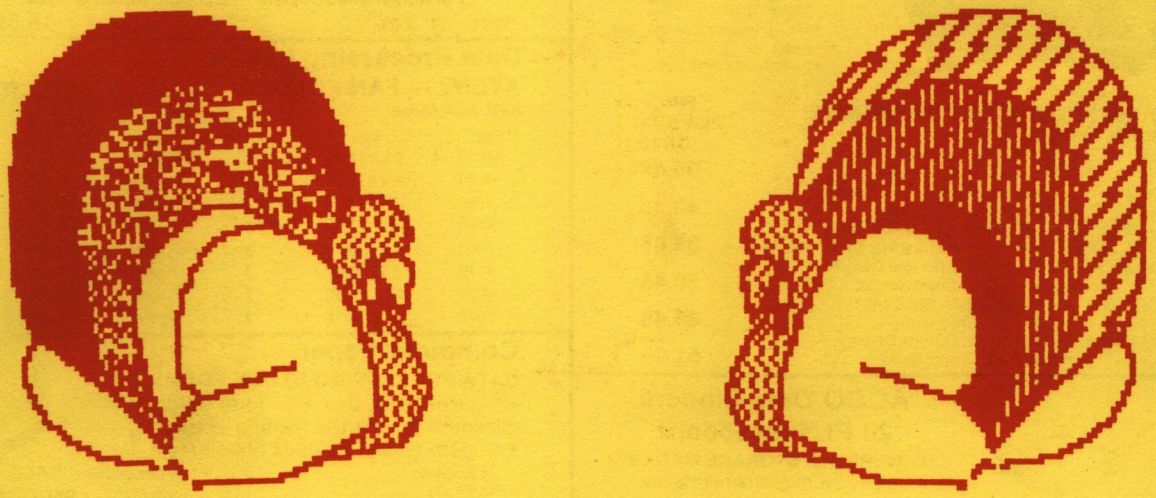
Apple PRESS

San Francisco Apple Core

IN THIS ISSUE

REM 1
Notices / NovDecJan 2
Call A.P.P.L.E. 3
B.A.S.I.C. - Joyce Conklin 4
FORTH - Ed Bronstein 11
Pascal: Quickie & A.S.E. -Max J. Nareff 12
Epson Salts - Michael Fischer 14
Easy Pascal I/O - Michael Fischer 18
Marketing - Don Scellato 25
Interest Groups 26
Repair Book, Review - Joyce Conklin 28
Club Offerings 30
Calendar 31



GENERAL MEETINGS:

INTEREST GROUPS NOV 19
GAMEFEST DEC 17

NOVEMBER, 1983

ARVEY PAPER & SUPPLIES

WE'RE CUTTING YOUR COS
ON COMPUTER and COPIER SUPPLIES

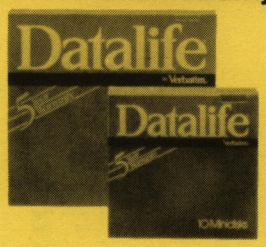
ARVEY offers you one stop shopping with thousands of items in stock, featuring daily savings on Office Supplies, Envelopes, Paper, Copier Supplies, Janitorial Supplies, and much, much more.

PLENTY OF FREE PARKING

These specials available at all 5 California locations listed below

SALE DATES: NOV 1 - NOV 30

Datalife. The name is the promise. The warranty is the proof.



Preserving your data is vitally important. Lose it and you've lost both time and money.

That's why it's vitally important to keep it on flexible disks that guarantee a longer life of trouble-free recording, storage and retrieval: Datalife® flexible disks with the 5-year warranty. That's 5 times longer than the industry standard. No wonder Verbatim is the standard of excellence in flexible disks, mini-disks, cassettes and minicassettes.

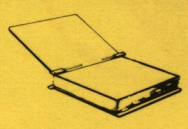
MODEL	DESCRIPTION	Price per Box of 10
MD 525-01	5¼" Single-Sided, Double Density, Soft Sectored	26.50
MD 525-10	5¼" Single-Sided, Double Density, Hard Sectored	26.50
MD 550-01	5¼" Double-Sided, Double Density, Soft Sectored	36.00
MD 577-01	5¼" Single-Sided, Quad Density, Soft Sectored	35.00
FD 34-9000	8" Single-Sided, Single Density, Soft Sectored	32.00
FD 34-8000	8" Single-Sided, Double Density, Soft Sectored	34.50

Dysan Diskettes

- 100% Surface tested
- Advanced burnishing techniques
- DY 10 tm lubricant for min. head use
- Auto load Certification

QUALITY WITHOUT COMPROMISE!

MODEL	DESCRIPTION	PRICE PER BOX OF 10
104/1D	5¼" Single Sided, Double Density, Soft Sectored, Unformated	36.45
104/2D	5¼" Doubled Sided, Double Density, Soft Sectored, Unformatted	43.45
107/1	5¼" Single Sided, Double Density, Hard Sectored, Unformatted, 10 Sectors	36.45
204/2D	5¼" Double Sided, Double Density, Soft Sectored, Unformatted, 96 T P I	50.45
3740/1D	8" Single Sided, Double Density, Soft Sectored, Unformatted	44.45
3740/2D	8" Double Sided, Double Density, Soft Sectored, Unformatted	51.95

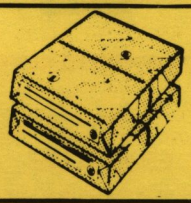


ACCO Data Binders 20 Pt. Pressboard

COVERS with STORAGE HOOKS

Coated for moisture resistance

No.	Size	Colors	Reg. Price	SALE each
54042	14⅞ x 8½ — unburst size	Lt. Blue	4.25	2.90
54070	14⅞ x 11 — unburst size	Blue, Green, Red	3.89	2.65
54270	11 x 14⅞ — burst size	Blue, Green, Red	3.00	2.42
23 pt. Accohide Cover — with storage hooks.				
56073	14⅞ x 11 — unburst size	Dark Blue	4.06	2.77



Xerographic Paper

For plain-paper copy machines

Boise Cascade O/D Xerographic or X9000

	Reg. Price	Price/ream of 500
8½" x 11" - 20	2.70	2.39
8½" x 14" - 20	3.68	3.19

3M Transparancy Film

For Plain Paper Copiers

100 sheets per box — black image on clear background

	Reg.	SALE
No. 501, 502, 503 — For Xerox and others	35.83	27.19
No. 506A — For Kodak Ectaprint	35.83	27.19

AVERY Self-Adhesive Copier Labels

For use in Xerox® and other sheet-fed plain paper copiers. Type master list. Load white label sheets into copier, and make copies of master list. For changes, change only the master. It's easy and inexpensive — sheet size 8½" x 11".

Number	Label Size	Labels per box	Reg Price	Sale Price
5351	1" x 2¾"	3300	19.24	15.59
5332	1" x 2¾"	8,250	39.60	31.82
5334	1" x 2¾"	16,500	71.40	57.39
For liquid toned copiers — Cannon, Saxon & Savin				
5331	1" x 2⅝"	3000	19.24	15.59

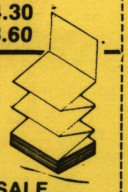
Data Processing Labels

AVERY — FAN FOLDED

Self Adhesive

No.	Size	Labels Across	Quantity Box	SALE per Box
4013	3½ x 15/16	1	5M	12.15
4014	4 x 1-7/16	1	5M	20.99
4031	3½ x 15/16	3	15M	36.49
4060	3½ x 1-7/16	1	5M	25.39
4033	4 x 1/7/16	3	15M	63.10
4015	5 x 15/16	1	5M	21.39
4030	3½ x 15/16	2	10M	24.30
4029	3½ x 15/16	4	20M	48.60

Computer Paper



DATA PROCESSING STOCK FORMS

- high bond quality • uniform weight and thickness for positive feeding everytime.
- smooth finish - extends ribbon life

Description single ply	Case quantity	SALE per case
9 1/2 x 11 - 15 lb - green bar or blank	3300	21.60
9 1/2 x 11 - 18 lb - green bar or blank	2600	18.90
9 1/2 x 11 - 20 lb - blank	2400	18.90
14 7/8 x 11 - 15 lb - green bar or blank	3500	31.95
14 7/8 x 11 - 18 lb - green bar or blank	2800	28.80
14 7/8 x 11 - 20 lb - blank	2600	30.00

SAN FRANCISCO
2275 Alameda St.
Tel. (415) 863-3664
Open Weekdays 8-5
Saturdays 9-2

OAKLAND
229 Castro St.
(Tel. (415) 839-8863
Open Weekdays 8-5:30
Saturdays 9-2

REDWOOD CITY
1445 Veterans Blvd.
Tel. (415) 367-0800
Open Weekdays 8-5:30
Saturdays 9-2

SAN JOSE
1381 North 10th St.
Tel. (408) 288-9280
Open Weekdays 8-5:30
Saturdays 9-2

SACRAMENTO
1101 Richards Blvd.
Tel. (916) 448-8400
Open Weekdays 8-5:30
Saturdays 9-2

CIDER PRESS

REM

A person learns some strange new ways after they live with a computer for a while. The relationships things bear to each other and the world around them start to take on new names and a vocabulary emerges as an expression or development of this new schema of perception. The change is slight at first, but soon *Weltanschauung* and *Ptolomeic* give way to *Hierarchical, Relational* and *Interrupt-based.*

Another symptom is that of a mutated orthography, with upper case letters popping up midWord. The words they pop up in are not really words, but a collection of letters each one of which holds some relation to the general subject. That is called ABSCAM - Acronym Based Speech Construction & Agglutinative Metaphor.

Aren't computers neat! Once I thought REM was something you did while you were asleep - then again, maybe it still is.

-ed.

THE CORE'S CORE

DIRECTORS:

Sami Capron	415/388-8631
Joyce Conklin	415/341-6955
Michael Fischer	415/933-4435
Gary Knox	415/521-0603
Don Scellato	415/672-2051
Alan L. Smerican	415/566-2342
Allan B. Woods	415/687-9659

MEMBER SERVICE/COMMITTEES:

Membership		
Alan L Smerican		415/566-2342
	Ted Chatterton	
Cider Press -		
Editor	Allan Woods	415/687-9659
BASIC	Joyce Conklin	415/341-6955
Int. Grp. Coord.	Sami Capron	415/388-8631
Marketing Dir.	Don Scellato	415/672-2051
Review Group	("Woody") Liswood	415/585-6375
General Meeting	Michael Fischer	415/933-4435
Treasurer	Gary Knox	415/521-0603
Coordinating Dir.	Joyce Conklin	415/341-6955

Computer Graphics . . . . Mei-Ying Chen
. . . Don

CIDER PRESS POLICY

All manuscripts, photographs, and other materials will be considered submitted free and released for publication. They become the property of the San Francisco Apple Core and the CIDER PRESS. Authors should clearly mark all material submitted for publication so that proper credit can be given.

The views expressed in any articles are those of the authors, and not necessarily those of the San Francisco Apple Core, Inc., the Editor, Club Officers or Directors . . . or anyone else in their right minds.

The San Francisco Apple Core Incorporated is a non - profit organization comprised of and supported by owners and users of Apple I I and Apple / / / computers. The Club is run by officers and committees, and the Club endeavors to aid Apple owners through its Program Libraries, Program Disks, meetings, and this publication.

Article Submission: The CIDER PRESS welcomes any and all articles dealing with Apple computers and their associated hardware and software. Articles may be submitted (typed or printed, double - space), or as disk text files using any of the popular word processors or text editors. Any author who has an article published in the CIDER PRESS will receive a free copy of the most current Disk - of - the - Month.

Reprints: All articles appearing in the CIDER PRESS not copyrighted by the author may be reprinted by other *non - profit* Apple User Groups, without prior permission, so long as proper credit is given to both the author and the CIDER PRESS. Permission to reprint a copyrighted article must be obtained in advance, by writing to the authors, c/o the San Francisco Apple Core.

San Francisco Apple Core
1515 Sloat Blvd.
San Francisco, CA 94132

INFO WANTED

Does anyone know anything about simulation programs for stock, commodity and/or gambling investment systems?

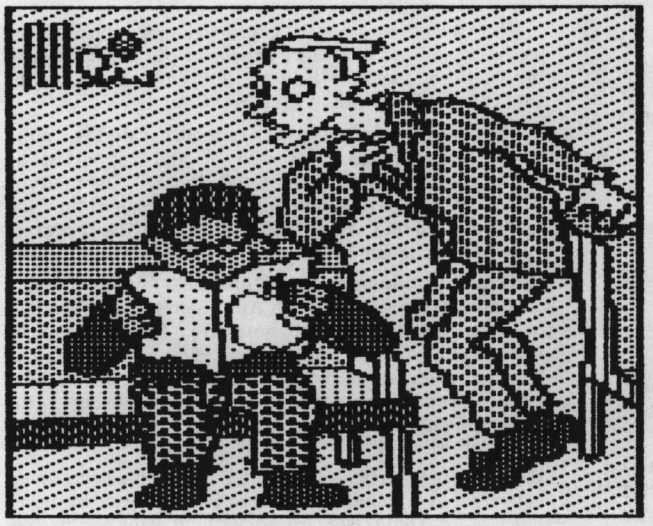
Please contact:

Joel Brodsky
5406 Telegraph Avenue
Oakland, CA 94609

(415) 655-1255

Personals

Ocassional programmer seeking a little sub-routine to call from time to time and exchange variables. Usual protocols. Parameters to suit. Reply Box "&" - Clubhouse.



NovDecJan

Interest Group Extravaganza for November's Meeting

November's general meeting, set for November 19 at 10:00 in Room 100, Building C, Fort Mason, will feature the interest group extravaganza. Each interest group -- Pascal, Printer, Basic, Forth, CP/M, Graphics, Telecommunications and others -- will give a short demonstration and have members on hand to answer questions you may have. If you have ever wanted to know about a specific interest group, or if you want to find out about how to join a group, the November meeting should be for you.

November's meeting will also be the place for you if you wish there was an interest group on a particular subject, or an interest group in your geographical area. Come and find out what additional benefits the San Francisco Apple Core can provide for you.

Games Festival in December

Do you like computer games? Would you like to try your hand at some of the newest and most spectacular games available for the Apple? Would you like to win a game, or purchase it at a discount? The come to the meeting on December 17.

From 6 to 10 game software companies will be featured at the December meeting. Members will have the opportunity to try each game being displayed. Each company will award one copy of its game by high score, random selection, or some other means. And the featured game, as well as other games sold by the companies, will be available for sale, some at a substantial discount. It's the perfect place for last minute Christmas shopping.

January Surprises

Apple Computer Company's general shareholders' meeting is scheduled for January, and speculation is running high that the MacIntosh will be announced at that meeting. If it is, we hope to have that machine at our January meeting for you to see. If not, we will probably have word about PRO-DOS, Apple's new disk operating system expected to be released publicly at that time. We will have a knowledgeable speaker at that meeting.

By Special Arrangement

Members of

The San Francisco Apple Core

May Join

Call A.P.P.L.E.

and Save $20.00 *on the First Year Initiation Fee.*

To take advantage of this limited time offer send the information asked for below along with a check made out to S.F.A.C. for $25.00

Your name and authorization will be forwarded to Call A.P.P.L.E. and your 12 month membership will begin.

Name:_____

Address:_____

City/State:_____ Zip:_____

S.F.A.C. MEMBERSHIP NUMBER!_____

*Your number can be found on your membership card – – and on your Cider Press mailing label.

B*A*S*I*C

Copyright (c) 1983 by Joyce Conklin

The first item this month is a correction and a complete explanation of the Ctrl-S problem discussed in the August issue. When I was ready to type the column for October, I couldn't find the letter which had been sent and also did not correctly remember the name of the helpful SFAC member who sent me the letter! It was not "Walt", but rather J. Morris Prosser, of Pebble Beach, CA. The answer he gave is as follows:

"The subroutine that handles this function is VIDWAIT, in the Autostart Monitor at $FB78. This subroutine is reached only when printing to the screen, and it checks for CTRL-S only after a RETURN. The reason the particular example you gave acts the way it does is that when Applesoft encounters a comma in a PRINT statement, it checks the position of the cursor.

If the cursor is at column 24 or greater, a carriage return is issued, otherwise the cursor is moved 16 places right and the next item is printed. In the case of items separated by semicolons, no carriage returns are issued, so the VIDWAIT routine never checks for CTRL-S.

In a nutshell, then, any program (not only Applesoft) that is printing to the screen can be interrupted by CTRL-S, if carriage returns are being printed."

Many thanks to Morris for his help. Also to Leigh Wilson who gave me a verbal answer to my question of several months ago about the ESC- function in PLE only working sometimes. It seems that the villain in this piece is the use of PR#1, which says "check for DOS commands and then for printer commands - only if it is neither of these then use the PLE commands" -- or something to that effect; I'm not sure that the dialect has been anthropomorphized correctly!

. . . . .

A rather interesting graphics program introduced to me by one of my students is included here for your edification and delight. Play around with the values of M and N (in line 150) and see what kinds of patterns you get. You could also experiment with the color, if you have a color monitor; HIRES color 3 is white, which will work best with a monochrome monitor.

```
90 REM "APPLE ROSE"
95 REM CONTRIBUTED BY RICH GOLDSTEIN
100  HOME
120  HCOLOR= 3
140  PRINT "ENTER M,N "
150  INPUT M,N
170  HGR2 : HPLOT 140,95
190  FOR A = 0 TO 256
     STEP  RND (1) + .01
210 R =  SIN ((M / N) * A) * 95
230  HPLOT  TO 140 + R *  SIN (A),
     95 + R *  COS (A)
240  NEXT A: GOTO 140
250  END
```

. . . . .

The flow chart which I developed for the Craps program appears in this issue, although not right here, because of the space requirements. The coding which was included in the October issue covers only the first four symbols (START through PAUSE) and the subroutine on instructions numbered 2000 and following.
The rectangular symbol with the extra vertical bars indicates that the detailed flowchart for that subroutine is elsewhere, not at that point in the main chart. My recollection of the betting procedure in craps is that you put your money down at the beginning of a roll and hope that the shooter doesn't blow it ("it" being whatever way you've bet, COME or DON'T COME or whatever!)

But it isn't necessary to continue to bet with each roll, while the shooter is trying to get his point. Notice that one "money" routine will work for all the times you need to calculate amounts; IT=1 is only set the first time through, when you decide if you want to bet at all, so you only get the message that you have $1000 at the beginning of the game. Every time you want to bet, the program must compare what you bet with

what you have and deliver the appropriate raspberry if you try to bet with the house's money! This comparison can also be in a subroutine, called from within the main money subroutine.

Speaking of subroutines, those which you access often during a progam should be located at the front - this will speed up the execution of the program - so maybe we had better renumber the instruction module from last month to 20000 (rather than 2000) as it will only be called once, or maybe not at all. Then line 20050 should read:

```
20050 RETURN
```

which will take you back to the part of your program where you ask if the user wants the instructions (Symbol #5 on the flow chart.) We now have line numbers from 2000 to use for the money subroutine(s), which will be used often during the course of the program. The coding (at least, what I have figured out tonight!) is as follows:

```
1999 REM SUBROUTINE TO FIGURE BETTOR'S
$$

2000 IF IT <> 1 THEN 2100

2010 PRINT "YOU HAVE $1000 TO START
WITH."

2005 M = 1000

2020 INPUT "HOW MUCH DO YOU WANT TO
     BET?"; A

2030 GOSUB 2500

2100 IF W <> 1 THEN 2200 : REM W is the
     flag for a winning roll

2110 M = M + A

2120 W = 0 : REM  Reset counter, ready
     for next roll

2130 RETURN

2200 M = M - A

2210 IF M > 0 THEN RETURN

2220 PRINT "SORRY, YOU ARE NOW BROKE!!"

2230 PRINT "COME BACK WHEN YOU HAVE MORE
     MONEY!"

2240 END

2499 REM COMPARISON MONEY SUBROUTINE

2500 IF A < M THEN RETURN

2510 PRINT "YOU CAN'T BET MORE THAN
     YOU HAVE!"

2520 PRINT "WHAT DO YOU WANT TO BET? IT
     BETTER BE LESS THAN $";M;" !!"

2530 INPUT A : GOTO 2500
```

You will notice that, with the exception of the title screen which was printed in the October issue, no effort has been made yet to have the screens all work properly. I like to work on the logic first to make sure that the program will RUN - then make the screens appear the way I want them. That's my contribution for the month - are you still working on yours?

R. J. Brooks of Berkeley, CA, sent in the following program; he called it "Quick and Dirty" but I think it is really quite nice - graphics dice and all. Notice that he has used the Ctrl-J in his REM lines to set them off from the body of the program.

Here's another idea for you to chew on over the month - maybe you'll be able to synthesize several ideas into one program good enough to submit the next time we have a programming contest! Notice that R. J. used the placement of the subroutines as I mentioned above, including the title screen.

```
10  REM  CRAPS
11  REM VERSION 1.1
12  REM  SFAC BASIC PROBLEM 9/83
13  REM
          ROLL TWO DICE

14  REM 7 OR 11 ON FIRST ROLL = WIN
15  REM 2,3,OR 12 ON 1ST ROLL = LOSE
16  REM ANY OTHER # = YOUR POINT TO MAKE
17  REM A 7 BEFORE MAKING POINT = LOSE
18  REM SHOW THE DICE ROLLING
19  REM KEEP SCORE ($1000) OF BETS
20  REM


30 TRUE = 1:FALSE = 0
40 CRAP = 7
50 KS =  - 16368:KD =  - 16384
80  DIM RT(6)
82  DIM L(21,2): REM  CORNERS
83  DIM S(6,24): REM SPOT OFFSETS
90  GOSUB 12000: REM  INSTRUCTIONS
92  GR : PRINT : PRINT : PRINT : PRINT
98  GOTO 10000
99  REM


100  PRINT "**************************
          ***********"

(There are thirty-eight stars in the
line
above.)

108  RETURN : REM
```

```
  110  PRINT "**
                    **"

  (There are thirty-four spaces between
   the left and right pairs of stars.)

  120  RETURN : REM


  200  REM

<<DISPLAY ROLL>>
  216  GOTO 239
  217  GR
  218  COLOR= 15
  219  FOR I = 1 TO 8 STEP 4
  220 X1 = L(I,1):X2 = L(I + 1,1)
  221 X3 = L(I + 2,1):X4 = L(I + 3,1)
  222  FOR Y = L(I,2) TO L(I + 4,2)
  223  HLIN X1,X2 AT Y
  224  HLIN X3,X4 AT Y
  225  NEXT
  226  NEXT
  228  COLOR= 0
  229  RETURN
  230  FOR I = 1 TO 0 * 2 STEP 2
  231  SPEED= 1
  232  PLOT S(0,I) + K,S(0,I + 1): PLOT
       S(0,I) + K,S(0,I + 1) + 1
  233  FOR X = 1 TO 25
  234 XX =  PEEK ( - 16336)
  235  NEXT X
  236  NEXT
  237  SPEED= 255
  238  RETURN
  239  GOSUB 217
  240 O = M:K = 1: GOSUB 230
  241 O = N:K = 22: GOSUB 230
  250  RETURN
  300  REM  <<DISPLAY POINT>>
  310  PRINT "YOUR POINT IS ";PO
  320  RETURN : REM


  330  DATA  1,1,17,1,22,1,38,1
  331  DATA 1,26,17,26,22,26,38,26
  332  DATA  8,12
  333  DATA  3,3,13,21
  334  DATA  3,3,13,21,8,12
  335  DATA   3,3,13,3,3,21,13,21
  336  DATA   3,3,13,3,3,21,13,21,8,12
  337  DATA 3,3,13,3,3,21,13,21,3,12,13,12
  400  FOR I = 1 TO 8: FOR J = 1 TO 2:
       READ L(I,J): NEXT : NEXT
  402  FOR I = 1 TO 6
  404  FOR J = 1 TO I * 2
  406  READ S(I,J): REM  SPOT OFFSETS
  408  NEXT : NEXT
  410  RETURN
  1000  REM

  <<ROLL 'EM>>
  1010  PRINT "STAKE = ";STAKE;
  1011  PRINT " WINNINGS = ";STAKE - 1000
  1020  INPUT "ENTER YOUR BET (TO QUIT
        BET 0) ";BET
  1030  IF BET < 1 THEN  END
  1032  IF BET > STAKE THEN  PRINT
        CHR$ (7)"SORRY, NO CREDIT !":
        GOTO 1020
  1040 J =  INT ( RND (1) * 6) + 1:K =
       INT ( RND (2) * 6) + 1
  1050 ROLL = J + K
  1060 RT(J) = RT(J) + 1:RT(K) = RT(K) +
       1: REM  RUNNING TOTALS
  1070 M = J:N = K: GOSUB 200: REM DISPLAY
  1080  GOSUB 3000: REM  TEST ROLL
  1100  IF WIN OR LOSE THEN  RETURN :
        REM  TO MAIN PROGRAM
  1110 FIRST = 0
  1120  GOSUB 300: REM  DISPLAY POINT
  1130  GOTO 1020
  1200  REM


  3000  REM
<< TEST ROLL >>
  3010  IF FIRST = 0 THEN 3200
  3020  IF ROLL = 7 OR ROLL = 11 THEN WIN
        = TRUE: RETURN
  3030  IF ROLL = 2 OR ROLL = 3 OR ROLL =
        12 THEN LOSE = 1:WIN = 0: RETURN
  3040 FIRST = 0:PO = ROLL: RETURN
  3200  REM  NOT FIRST ROLL SO COMPARE
        TO POINT
  3210  IF ROLL = PO THEN WIN = 1: RETURN
  3230  IF ROLL = 7 THEN LOSE = 1: RETURN
  3250 WIN = 0:LOSE = 0: RETURN
  3500  PRINT "YOU'RE A WINNER !"
  3505  PRINT  CHR$ (7)
  3507 WIN = 0:LOSE = 0:PO = 0:FIRST = 1
  3510  RETURN
  3600  PRINT "YOU LOSE !"
  3605  PRINT  CHR$ (7)
  3607 LOSE = 0:PO = 0:FIRST = 1
  3610  RETURN
  10000  REM

<<< MAIN PROGRAM >>>

  10004  GOSUB 400
  10006 FIRST = 1:WIN = 0:LOSE = 0:PO = 0
  10008 STAKE = 1000
  10010  REM
  10020  GOSUB 1000: REM  ROLL 'EM
  10030  IF WIN = TRUE THEN STAKE = STAKE
         + BET: GOSUB 3500: GOTO 10050
  10036  GOSUB 3600
  10040 STAKE = STAKE - BET
  10050  IF STAKE > 0 AND STAKE < 2000
         THEN  GOTO 10020
  10060  IF STAKE =  > 2000 THEN  GOTO
         14000: REM   TABLE CLOSED !
  10070  GOTO 14900: REM   YOU LOSE!
  12000  REM
<< INSTRUCTIONS >>
  12100  GOSUB 100: GOSUB 110
  12110  PRINT "**             C R A P S
                    **"

  (There are twelve spaces before "CRAPS"
   and 13 after.)

  12111  GOSUB 110
  12120  PRINT "**          BY  R. J.
  BROOKS
                    **"
  12130  GOSUB 110: GOSUB 100
  12150  GOSUB 20000: REM PAUSE/SOUND
  12160  FOR I = 1 TO 24: PRINT : NEXT
  12200  PRINT "THIS IS A GAME OF CHANCE.
         YOU PIT YOUR"
```

```
12210  PRINT "STAKE ($1000) AGAINST
       LADY LUCK IN A"
12220  PRINT "ROLL OF THE DICE."
12222  PRINT
12230  PRINT : PRINT "IF YOU ROLL 7 OR
       11 ON THE FIRST TRY"
12240  PRINT "THAT'S A NATURAL AND YOU
       WIN YOUR BET."
12242  PRINT
12250  PRINT "IF YOU ROLL 2,3 OR 12 ON
       THE FIRST TRY"
12260  PRINT "THAT'S TOO BAD, YOU LOSE
       YOUR BET."
12262  PRINT
12270  PRINT "ANYTHING ELSE ON THE
       FIRST ROLL IS"
12280  PRINT "YOUR POINT. YOU THEN TRY
       TO ROLL YOUR"
12290  PRINT "POINT AGAIN BEFORE YOU
       'CRAP OUT' WITH"
12300  PRINT "ANOTHER 7. IF YOU CRAP OUT
       YOU LOSE"
12310  PRINT "YOUR BET."
12312  PRINT : PRINT
12320  INPUT "PRESS RETURN TO BEGIN...";
       TE$
12330  RETURN
12340  REM


14000  FOR I = 1 TO 3000: NEXT I
14004  FOR I = 1 TO 5: PRINT  CHR$ (7);:
       NEXT
14010  PRINT "THIS TABLE IS CLOSED !"
       CHR$ (7): PRINT "YOU HAVE WON ";:
       FLASH : PRINT "$"STAKE - 1000:
       NORMAL
14020  FOR I = 1 TO 3000: NEXT I
14030  PRINT : PRINT "IF YOU WANT AN
       ESCORT TO YOUR CAR"
14040  PRINT "PLEASE TELL THE CASHIER...
       GOOD NIGHT."
14050  END : REM  TABLE CLOSED


14900  FOR I = 1 TO 3000: NEXT I
14910  FOR I = 1 TO 3: PRINT  CHR$ (7);:
       NEXT
15010  PRINT : PRINT "YOU ARE BUSTED.
       GOOD NIGHT."
15020  END : REM  BUSTED


20000 SP = 49200
20010  FOR I = 1 TO 8
20020  FOR J = 1 TO 50
20029 X =  PEEK (SP)
20030  IF I < 5 THEN 20040
20032  IF I < 9 THEN Y = Y + 1:
       GOTO 20040
20040  NEXT
20050  NEXT
20060  RETURN


DID YOU GET ALL THAT?
```


for REV 7 or later Apple II+ keyboards.

The Invaluable
Useful
Convenient
Fully Transparent

SHIFT KEY MODIFICATION

Lets your SHIFT key function like a true shift key under – software control –

Does not interfere with any functions

Recognized by:

- *AppleWriter II*
- *Magic Window*
- *PIE:Writer*
- *Many others*

80 Column cards

- *Videx*
- *Sup'R Term*
- *etc.*

CP/M Software

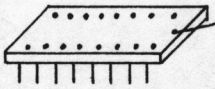
- *WordStar, etc.*

Your own programs!

High grade components
Instructions included
Full product support.

$7.50 + tax at meetings

$8.50 ppd mailorder



The San Francisco Apple Core
1515 Sloat Blvd., Suite 2
San Francisco, CA 94132

Fig. 1

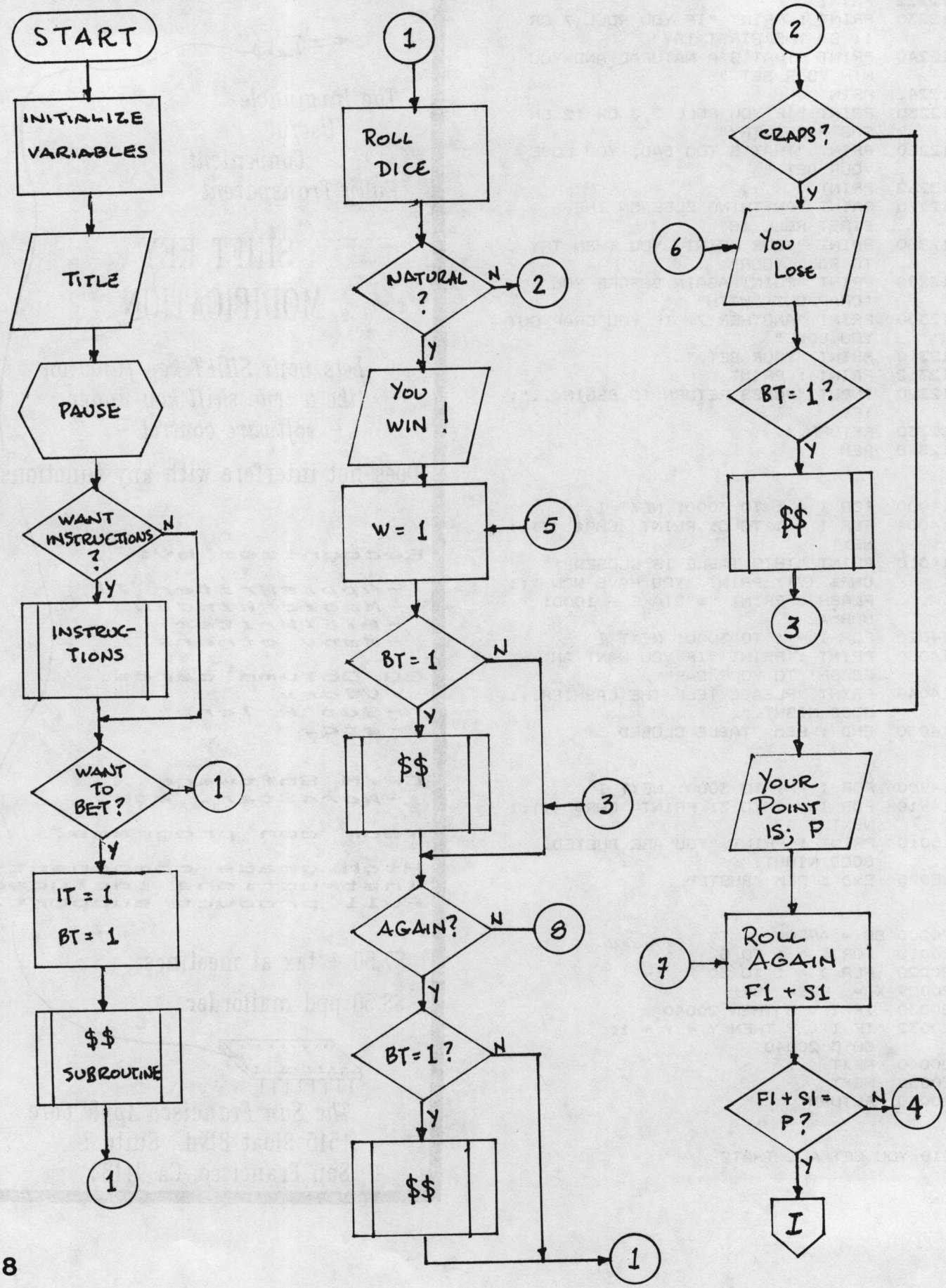
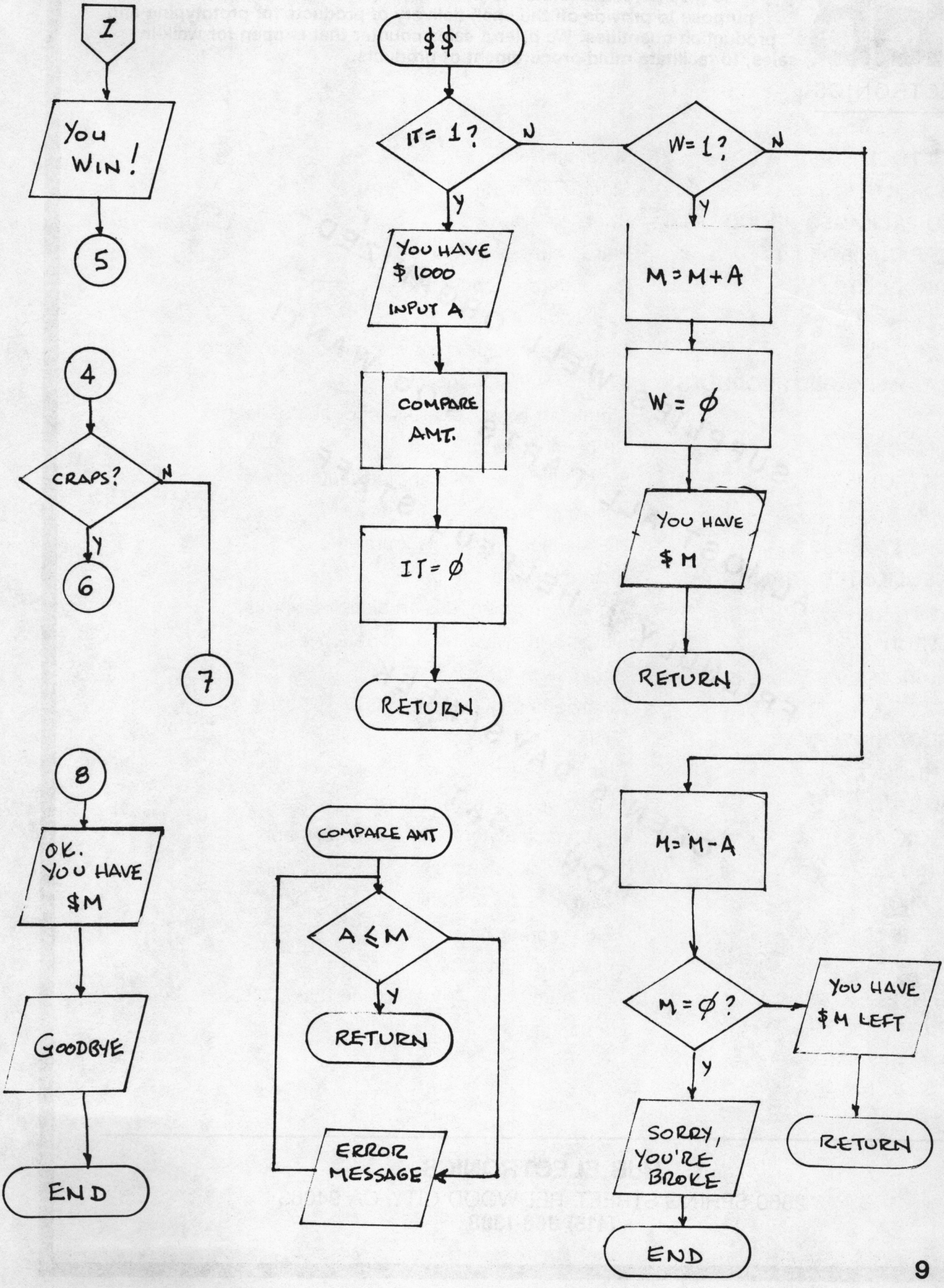

START
INITIALIZE VARIABLES
TITLE
PAUSE
WANT INSTRUCTIONS ?
N
Y
INSTRUC-TIONS
WANT TO BET?
1
Y
IT = 1
BT = 1
$$ SUBROUTINE
1
1
ROLL DICE
NATURAL ?
N
2
Y
YOU WIN !
W = 1
5
BT = 1
N
Y
$$
3
AGAIN?
N
8
Y
BT = 1?
N
Y
$$
1
2
CRAPS?
Y
6
YOU LOSE
BT = 1?
Y
$$
3
YOUR POINT IS; P
7
ROLL AGAIN F1 + S1
F1 + S1 = P?
N
4
Y
I

Fig. jam?


I
You Win!
5
4
CRAPS?
N
Y
6
7
8
OK. You have $M
GOODBYE
END
$$
IT = 1?
N
Y
You have $1000
INPUT A
COMPARE AMT.
IT = ∅
RETURN
COMPARE AMT
A ≤ M
Y
RETURN
ERROR MESSAGE
W = 1?
N
Y
M = M+A
W = ∅
You have $M
RETURN
M = M-A
M = ∅?
Y
SORRY, YOU'RE BROKE
END
You have $M left
RETURN

HdB ELECTRONICS

HDB Electronics is a small distributor that prides itself on the ability to give personal service to both large and small customers. It is our purpose to provide off the shelf delivery of products for prototyping and production quantities. We offer a sales counter that is open for walk-in sales, to facilitate rapid procurement of products.

ADEL TOOL	Metal nibblers
AKRO-MILS	Storage cabinets, plastic boxes
ALCO PACKAGED PRODUCTS	Switches, knobs
AMEREX/UNIBOX	Plastic chassis boxes
AP PRODUCTS	Protoboards, connectors, dip jumpers
ARCOLECTRIC	Switches
ARTEC	Circuit boards for prototype and custom designs
AUGAT PACKAGED PRODUCTS	IC and transistor sockets
BUD	Aluminum boxes, chassis, and rack panels
CAMBRIDGE	RF connectors
CHEMTRONICS	Electronic chemicals and desoldering wick
COLUMBIA	Wire and cable
CONNECTOR DIST.	"D" connectors, shells, hardware
CONSOLIDATED WIRE	Electronic wire and cable
DATAK	Transfer lettering, design aids, etch products
DELTRON	Linear and switching power supplies
DENNISON	Cable tie products
DURACELL	Alkaline and Ni-Cd batteries
EASCO/HUNTER	Tools for electronics
E-Z-HOOK	Electronic test clips and cables
HAMMOND	Plastic boxes, metal chassis
HICKOK	Digital multimeters and test equipment
ICO-RALLY	Heat shrink tubing, cable ties, hardware
INTERLOK	Collet knobs, cabinets, card frames
KESTER	Solder and flux
LITTELFUSE	Fuses and accessories
LMB HEEGER	Aluminum boxes, chassis, and rack panels
MACOM	RF connectors
MASTER	Heat guns

SUPPLIES WELL ORGANIZED-
ALMOST ALL PARTS YOU WANT!
FRIENDLY & HELPFUL STAFF
OPEN 6 DAYS/WEEK
MON - SAT

HdB ELECTRONICS
2860 SPRING STREET, REDWOOD CITY, CA 94063
(415) 368-1388

(415) 368-1388
HdB ELECTRONICS
(415) 368-1388

FORTH WORLD

-Ed Bronstein.

For those interested in learning some fundamentals of computers as well as learning to program in FORTH, a few articles will follow directed to both the beginner in computers and the beginner in FORTH.

FORTH is more than a computer language -- it is an operating system that can be used for any computer application. It is extensible -- meaning that if a command is not in a system when a user needs it, the command can be formulated and put into the system at will and then can be used at any time.

The power of FORTH is enormous in that it can accomplish more than any other language can using the same amount of memory. It runs programs 10 times faster than BASIC; it is more memory-conservative than an assembler and, although FORTH will run slightly slower than an assembler, programming itself will go something like 30 times faster!

The Apple Core now has available to members four FORTH diskettes. One is the source code (or "kernel") of a system written by George B. Lyons of Jersey City, New Jersey and donated to public domain in 1981. Two diskettes are utilities added to the Lyons system by the Apple Core FORTH group.

The fourth diskette is a FORTH system, a word processor based on the FORTH editor (this article is being written on the word processor), and a tutorial which attempts to lead the absolute beginner into a working relationship with FORTH.

This series of Cider Press articles on FORTH will, in many cases, go along with the word processor-tutorial diskette. To be certain of acquiring the tutorial, I believe the diskette will be labeled "BLOSSOMING FORTH".

FORTH was developed around 1970 at the National Radio Astronomy Observatory by Charles H. Moore who was looking for a better method than that which then existed for controlling the observatory's telescopes. FORTH worked so extremely well that other astronomers began to use it and FORTH has been attracting admirers ever since.

There are a number of publications now available for the dissemination of FORTH information. Probably the best instruction book to be had is STARTING FORTH by Leo Brodie. The FORTH tutorial on the Apple Core diskette goes along pretty well with this book.

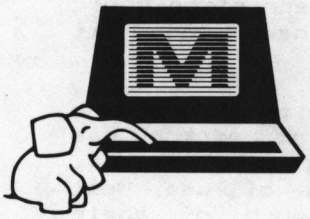
We won't go into programming in this article because if you're interested in FORTH you'd probably like to have a system for your Apple so that you can follow along with the instructions. So stop at the sales booth at the next core meeting and get the tutorial (and, perhaps, the other three FORTH diskettes).

AVAILABLE FOR CHARTER:

APPLE IIe Computers, moniters, printers, modems and other periferals as arranged. Charter from one to eight APPLE IIe computers for one to one hundred non-continuous hours. IBM PC and other models available for demonstration/sale.

Introductory Classes Available for children, adults and special interest groups. Special Christmas Vacation Program.

For Details Contact: THE MIGHTY COMPUTER CENTER, 5809 Geary Blvd., S.F. (415) 668-3888



THE MIGHTY COMPUTER CENTER

Pascalia and Fugue

–Max J. Nareff

Quickie

Possessors of hard-copy libraries of programs may find this little procedure useful.

Installed in any print-out or text formatting program, it will obviate the need to search through one's collection of diskettes, or to execute the fine CATALOG program appearing in the latest Pascal SFAC Disk.

Either the COPY function or the DELETE procedure may be used to isolate and write the volume name, which can serve as a program self-locator, whenever the copy is reviewed.

Beginners are referenced to page 24 of the Apple Pascal language reference manual for details of COPY and DELETE.

A number of print and text-formatting programs are present in the club Pascal library.

```
PROCEDURE VOLUMENAME;
   (*Declare vars..VOLNAME,
FILENAME:STRING;*)
BEGIN
   VOLNAME := FILENAME;
   VOLNAME :=
COPY (VOLNAME,1,(LENGTH(VOLNAME)-
((LENGTH(VOLNAME)-POS(':',
VOLNAME)))));
   WRITELN (VOLNAME);
   (*Alternate Mode *)
DELETE(FILENAME,POS(':',FILENAME)+1,
LENGTH (FILENAME)-POS(':',FILENAME));
   WRITELN (FILENAME)
END;
```

A. S. E.

Volition Systems' A(dvanced S(ystem E(ditor (ASE) was reviewed in the Cider Press, Nov./Dec.82. No description short of a minimanual could have done it justice.

Over a years experience with this superb enhancement of the p-System editor has heightened our appreciation of its scope, efficiency and its ease of use. Most users of Pascal spend much of their computer time working with the editor; Unquestionably, the ASE makes this easier, in fact a delight.

Volition has recently issued version 0.9 eliminating a few bugs, not encountered by this user. The ASE has been reduced in size, from 100 to 99 blocks, permitting a larger text buffer of 7K, with the swapping option on.

For those uncomfortable with the extra disk action that swapping entails, adjustments can be made. Unlike the earlier versions, these arcane subtleties are now pre-set. Additionally, a hard copy of all ASE commands and key definitions, over 60 in all, is provided.

Operations with ASE are simpler than this spate of keys implies, for most of the commands are those inherent in Pascal itself. It is the enhancements, however, which make the invigorated editor sparkle.

Most impressive about ASE is the mobility it provides for the cursor. Take a quick trip across the screen with the following commands:

Jump to end of line 6 below - 7L; Or to beginning of line 3 below - 4B; To the next word - W; Or 5 words along the line - 5W; Get a specific character - g; Again - G; Cursor to column 1 --<ESC> 0; Home - <ESC> H; Move part of line right or left - K + arrow; Move last line to top of screen - U; Or previous page - O.

Most of these commands can be invoked as well during X(change through the use of non-printing characters (ESC + case-insensitive key). The new version even permits use of the InsertCh and DeleteCh commands from the main editor prompt line, no longer are they confined in the X(change mode alone. In the "paint" mode, characters may be exchanged in any direction, useful for editing columns or making boxes.

Exceptionally useful is the ability to define up to 8 function keys or macros. Thes can be prepared in 3 different ways. Once generated, they can be made available across repeated editing sessions by means of a special marker, "$PROFILE", and use of embedded text forms.

Everytime the file is called for editing, the function keys are "taken-up" or activated and perform as programmed by typing <ESC> n (1..8). An example of a text form follows.

```
(*!X!*!F8!*!F7!*!F6!*!F5!*!F4!*!F3!*!F2!*!F1!.
{**********************************}!n!n!.
QU!.QB!.PROCEDURE
!.END!.READ!.WRITE!.BEGIN!N!s-!----!.
*).
```

When automatically taken up at the start of an edit session, the user no longer types out any of the above words, just <ESC + n>. ESC 1 types "BEGIN"; ESC 6 types "PROCEDURE", etc. ESC 7,QU,quits and updates the file, ESC 6,QB,quits, updates, and returns the file to the editor, very handy for the forgetful or the worrisome; ESC 8 draws a line with a blank line below.

Another valuable asset is the ability to utilize the filer without leaving the editor. Whenever a prompt calls for a file name, such as "C(opy from f(ile", a "?" will call-up the file menu of the specified disk, each file name preceded by a letter. A second "?" produces the first text line of each file listed on the screen, so that its contents may be identified. Typing the letter causes the file to copied into the current text.

The ability to use up to 26 markers in a single file makes it easy to copy multiple sections of many files into the file under edit. A few additional markers are reserved for ASE housekeeping, for log, tag, cursor and syntax error positions, etc. Most important is the "$PROFILE" marker,which when placed in front of an embedded text form will activate the stored function keys on editing initiation.

Efficiency is aided by being able to edit a file directly from any disk, converting the original file to "File.Back", so that the original version remains intact. A file being edited can be redirected to another location, even with another name. The "?" summons also operates from the E(dit promptline.

Files larger than the APPLEs' memory capacity are edited through ingenuous buffer management. Multiple files can be effortlessly edited in a single session, since one edit may be interrupted, to be replaced by another, and another - this is called "nesting". Nested files can be chained. Anything the basic p-System can do, the ASE will do and much more.

One source of discomfort: The large size of the editor, and the recommendation to use swapping-on when dealing with larger files, make for considerable disk activity and a significantly sluggish response to commands. However, this can be minimized by use of a RAM disk.

There are several excellent solid state disks available and their use is strongly recommended. This observer uses Synetixs' FLASHDISK, previously reviewed in the newsletter. The ASE is promptly transferred to the SSD on boot, and then it's off to the races.

The ASE is a valuable and enjoyable tool. One becomes addicted to its facilities. No serious Pascal user should be without it.

More of the many intriguing features of the ASE will be the subject of the next revisit.

Volition Systems, Inc. P.O.Box 1236, Del Mar ,Ca 92014 Synetix MicroProducts 15050 N.E. 95th. Redmond, Wa 98052

SUBSCRIPTIONS

For IAC Club Members Only

INTERNATIONAL APPLE CORE™

908 GEORGE STREET, SANTA CLARA, CALIFORNIA 95050

☐ 9 issues for only $19.50
($29.25 at newsstands)
$17.00 for members of IAC clubs only

☐ Foreign: 9 issues
$36.00 Surface
$29.00 for members of IAC clubs only

☐ Payment enclosed
(US funds only)

Start my subscription with issue # _____

Name _____

Address _____

City _____ State _____ Zip _____

☐ Name of Club _____ Signature of club officer _____
Valid only with signature

InterBank # ☐☐☐☐ 4 digit number above your name on MasterCard™ ☐ VISA

Charge Card # ☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐

_____
Credit Card Signature Card Expires

Dumping It

Epson Salts

(C) 1983 by Michael Fischer

Last of a Series

There are three ways (more actually) to enter the screen dump routine introduced by this series into your Apple.

After the program is on disk just BRUN TEXT SCREEN DUMP (or whatever other name you have called it), and, unless you turn off the computer, load another file into the area (memory locations 768-918; $300-396) or mess up the pointers, just type control-P any time you want a text screen printed on your printer. If the points do get messed up, then CALL 768 from BASIC to help clear things up.

The three ways to enter the program are:

1. If you have an assembler such as Lisa, Big Mac, Apple Tool Kit, SC Assembler, ORCA/M or Merlin then enter the material found in listing 1 using the editor of your assembler and save the object code file on disk.

2. If you don't have an assembler but have either an Apple IC with Integer BASIC in ROM or on a ROM card, or have a langauge card with INTBASIC loaded on it (to do this run HELLO on the DOS 3.3 System Master), theny you can use the mini assembler. Follow these steps:

a. Get into Integer BASIC (prompt character ">"). Usually the command INT will do this if you are not already in Integer BASIC.

b. Enter the monitor by typing CALL -151. The prompt character should be "*".

c. From the monitor type F666G. The prompt character should be "!".

d. Type each line in listing 2. For each line after the first, type a space before entering the line. Press return at the end of each line.

e. When you are done type $FF69G to put you back in the monitor. Then type 38D:00 28 50 04 04 04 80 A8 D0 07 07 07.

Next type 3D0G to put you back in Integer BASIC. Finally type BSAVE TEXT SCREEN DUMP, A$300, L$97.

3. If you have neither an assembler or the mini-assembler:

a. Type CALL -151 to enter the monitor.

b. Enter each line in listing 3, followed by a return.

c. Type 3D0G to get back into BASIC.

d. Type BSAVE TEXT SCREEN DUMP, A$300, 1$97.

That's all there is to it. Listing 4 is how the assembler output of the program looks after being assembled by my assembler.

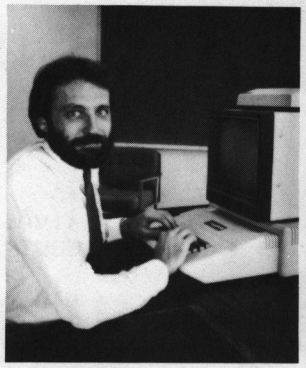
There is another way to get the program. Come to the Epson Interest Group meeting. We meet on the fourth Tuesday of each month at 7:30 p.m. at the Apple Core Clubhouse, 1515 Sloat Blvd., San Francisco.

LISTING 1.

```
 1  IVAL      EQU $ED
 2  ILIMIT    EQU $F9
 3            ORG $300
 4  SETHOOKS  LDA #<ROUTINE
 5            STA $38
 6            LDA #>ROUTINE
 7            STA $39
 8            JMP $3EA
 9  ROUTINE   JSR $FD18
10            CMP #$90
11            BNE OUT
12            JSR SCRNDUMP
13 OUT        RTS
14 SCRNDUMP   LDA #$01
15            JSR $FE95
16            LDA #$1B
17            JSR COUT
18            LDA #$40
19            JSR COUT
20            LDX #0
21 MAINLOOP   LDA ISTLO,X
22            STA IVAL
23            LDA ISTHI, X
24            STA IVAL+1
25            LDA ILIMLO,X
26            STA ILIMIT
27            LDA ILIMHI,X
28            STA ILIMIT+1
29 LOOP       JSR DOLINE
30            CLC
31            LDA #128
32            ADC IVAL
33            STA IVAL
34            LDA #0
35            ADC IVAL+1
36            STA IVAL+1
37            CMP ILIMIT+1
38            BLT LOOP
39            BEQ C1
40            JMP NEXTI
41 C1         LDA IVAL
42            CMP ILIMIT
43            BLT LOOP
44            BEQ LOOP
45 NEXTI      INX
```

"IDS helped me invent a new kind of program.

"And I'm not even a programmer."



— Lew Newman
Senior Vice President
Strategy Wares, Inc.
McLean, Virginia

Here's how IDS saves you time and money in creating your application programs:

Now, you can avoid buying expensive special purpose software packages. IDS makes it easier than ever before to write your own, using Applesoft BASIC — the native language of the Apple II. Just take advantage of these features:

- **Sophisticated screen formatting** to give you capabilities previously found only in high-priced mainframe software packages.
- **Simplified disk file I/O procedures and record format definition** that streamline your efforts.
- **Full printline formatting capability** (including $ and zero suppress) that lets you create professional, easy-to-read output quickly and simply.
- **Special editor** that lets you predefine data structures — screens, disk records and printlines — in terms of **Applesoft variables**, making IDS easy to use in your programs.
- **Machine-language subroutines** that let you access your data structures from Applesoft using a very convenient interface.
- **A convenient library** for maintaining data structures.
- **Excellent documentation with tutorial** to guide you.

In short, IDS means that you don't have to reinvent the wheel every time you create a program. And IDS opens new doors for you. You can do I/O programming yourself. Even novice programmers can follow our documentation and tutorial in defining input and output specifications.

And with IDS, your program is halfway completed and tested — **before you've typed a single line of code.**

All of these time and money saving benefits are yours because IDS — the Integrated Development System — is the only easy-to-use, fully integrated combination of Applesoft enhancements. That's why Lew Newman and others have found in it a software tool unlike any they have ever known.

Here's what software professionals and users have been saying about IDS:

"Every serious Applesoft programmer should have a copy of IDS . . . (It is) a fine piece of work . . .a very good piece of packaging."

— Doug Carlston
President,
Broderbund Software

"This (IDS) is a beautiful and elegant product!"

— C. Keith Conners, Ph.D.
Director of Research,
Dept. of Psychiatry
Childrens Hospital National
Medical Center
Washington, D.C.

"One of Applesoft's greatest deficiencies is in the area of formatted print. IDS remedies this through definition of print formats."

— David Morganstein
Softviews,
Washington Apple Pi

SPECIAL INTRODUCTORY PRICE: ONLY $85!

If IDS does not do everything we say, return the package within 30 days for a full refund.

IDS works with Applesoft BASIC on 48k Apple II Plus and Apple IIe computers with DOS 3.3. The 16k language card is supported. Apple and Applesoft are registered trademarks of Apple Computer, Inc.

Improve your productivity today!

Get IDS now!

Call (703) 471-0328

or

Write us at the address below. VISA and Mastercard orders accepted.

IDS from

r.r. michaels, inc. 7 E. Loudoun St., Leesburg, Virginia 22075

```
46              CPX #3
47              BLT MAINLOOP
48              LDA #0
49              JSR $FE93
50              JMP $FD6F
51              RTS
52 DOLINE       LDY #0
53 LINELOOP     LDA (IVAL),Y
54              JSR COUT
55              INY
56              CPY #40
57              BLT LINELOOP
58              JSR CROUT
59              RTS
60 COUT         STA $C090
61 GOTIT        LDA $C1C1
62              BMI GOTIT
63              RTS
64 CROUT        LDA #$0D
65              JSR COUT
66              LDA #$0A
67              JMP COUT
68 ISTLO        DFB $00,$28,$50
69 ISTHI              DFB $04,$04,$04
70 ILIMLO       DFB $80,$A8,$D0
71 ILIMHI       DFB $07,$07,$07
```

LISTING 2.

```
300:LDA #$0B
 STA $38
 LDA #$03
 STA $39
 JMP $3EA
 JSR $FD18
 CMP #$90
 BNE $315
 JSR $316
 RTS
 LDA #$01
 JSR $FE95
 LDA #$1B
 JSR $37A
 LDA #$40
 JSR $37A
 LDX #0
 LDA $38D,X
 STA $ED
 LDA $390,X
 STA $EE
 LDA $393,X
 STA $F9
 LDA $396,X
 STA $FA
 JSR $36A
 CLC
 LDA #$80
 ADC $ED
 STA $ED
 LDA #0
 ADC $EE
 STA $EE
 CMP $FA
 BCC $33B
 BEQ $354
 JMP $35C
 LDA $ED
 CMP $F9
 BCC $33B
 BEQ $33B
 INX
 CPX #$3
 BCC $327
 LDA #$0
 JSR $FE93
 JMP $FD6F
 RTS
 LDY #$0
 LDA ($ED),Y
 JSR $37A
 INY
 CPY #$28
 BCC $36C
 JSR $383
 RTS
 STA $C090
 LDA $C1C1
 BMI $37D
 RTS
 LDA #$0D
 JSR $37A
 LDA #$0A
 JMP $37A
```

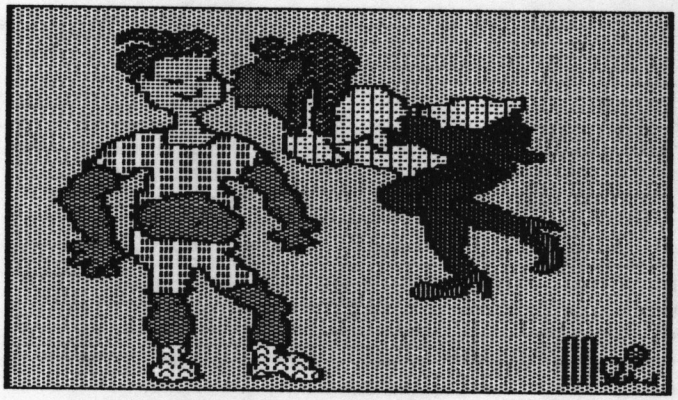
LISTING 3.

```
0300: A9 0B 85 38 A9 03 85 39 4C EA 03
030B: 20 1B FD C9 90 D0 03 20 16 03 60
0316: A9 01 20 95 FE A9 1B 20 7A 03
0320: A9 40 20 7A 03 A2 00 BD 8D 03
032A: 85 ED BD 90 03 85 EE BD 93 03
0334: 85 F9 BD 96 03 85 FA 20 6A 03 18
033F: A9 B0 65 ED 85 ED A9 00 65 EE
0349: 85 EE C5 FA 90 EC F0 03 4C 5C 03
0354: A5 ED C5 F9 90 E1 F0 DF E8 E0 03
035F: 90 C6 A0 00 20 93 FE 4C 6F FD 60
036A: A0 00 B1 ED 20 7A 03 C8 C0 28
0374: 90 F6 20 83 03 60 8D 90 C0
037D: AD C1 C1 30 FB 60 A9 0D 20 7A 03
0388: A9 0A 4C 7A 03 00 28 50 04 04 04
0393: 80 A8 D0 07 07 07
```

LISTING 4.

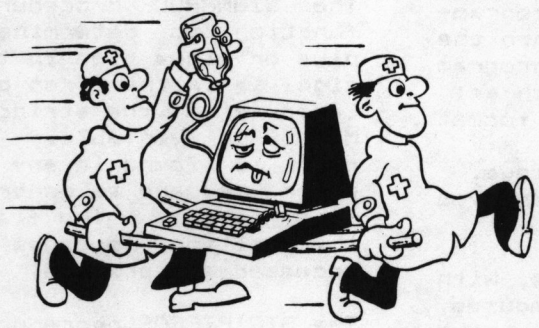
```
                    1  IVAL      EQU $ED
                    2  ILIMIT    EQU $F9
                    3            ORG $300
0300: A9 0B         4  SETHOOKS  LDA #<ROUTINE
0302: 85 38         5            STA $38
0304: A9 03         6            LDA #>ROUTINE
0306: 85 39         7            STA $39
0308: 4C EA 03      8            JMP $3EA
030B: 20 1B FD      9  ROUTINE   JSR $FD18
030E: C9 90        10            CMP #$90
0310: D0 03        11            BNE OUT
0312: 20 16 03     12            JSR SCRNDUMP
0315: 60           13  OUT       RTS
0316: A9 01        14  SCRNDUMP  LDA #$01
0318: 20 95 FE     15            JSR $FE95
031B: A9 1B        16            LDA #$1B
031D: 20 7A 03     17            JSR COUT
0320: A9 40        18            LDA #$40
0322: 20 7A 03     19            JSR COUT
0325: A2 00        20            LDX #0
0327: BD 8D 03     21  MAINLOOP  LDA ISTLO,X
032A: 85 ED        22            STA IVAL
032C: BD 90 03     23            LDA ISTHI, X
032F: 85 EE        24            STA IVAL+1
0331: BD 93 03     25            LDA ILIMLO,X
0334: 85 F9        26            STA ILIMIT
0336: BD 96 03     27            LDA ILIMHI,X
0339: 85 FA        28            STA ILIMIT+1
033B: 20 6A 03     29  LOOP      JSR DOLINE
033E: 18           30            CLC
033F: A9 B0        31            LDA #128
0341: 65 ED        32            ADC IVAL
0343: 85 ED        33            STA IVAL
0345: A9 00        34            LDA #0
0347: 65 EE        35            ADC IVAL+1
0349: 85 EE        36            STA IVAL+1
034B: C5 FA        37            CMP ILIMIT+1
034D: 90 EC        38            BLT LOOP
034F: F0 03        39            BEQ C1
0351: 4C 5C 03     40            JMP NEXTI
0354: A5 ED        41  C1        LDA IVAL
0356: C5 F9        42            CMP ILIMIT
0358: 90 E1        43            BLT LOOP
035A: F0 DF        44            BEQ LOOP
035C: E8           45  NEXTI     INX
035D: E0 03        46            CPX #3
```

```
035F: 90 C6     47              BLT MAINLOOP
0361: A0 00     48              LDA #0
0363: 20 93 FE  49              JSR $FE93
0366: 4C 6F FD  50              JMP $FD6F
0369: 60        51              RTS
036A: A0 00     52 DOLINE       LDY #0
036C: B1 ED     53 LINELOOP     LDA (IVAL),Y
036E: 20 7A 03  54              JSR COUT
0371: C8        55              INY
0372: C0 28     56              CPY #40
0374: 90 F6     57              BLT LINELOOP
0376: 20 83 03  58              JSR CROUT
0379: 60        59              RTS
037A: 8D 90 C0  60 COUT         STA $C090
037D: AD C1 C1  61 GOTIT        LDA $C1C1
0380: 30 FB     62              BMI GOTIT
0382: 60        63              RTS
0383: A9 0D     64 CROUT        LDA #$0D
0385: 20 7A 03  65              JSR COUT
0388: A9 0A     66              LDA #$0A
038A: 4C 7A 03  67              JMP COUT
038D: 00 28 50  68 ISTLO        DFB $00,$28,$50
0390: 04 04 04  69 ISTHI        DFB $04,$04,$04
0393: 80 A8 D0  70 ILIMLO       DFB $80,$A8,$D0
0396: 07 07 07  71 ILIMHI       DFB $07,$07,$07

--End assembly--
153 bytes
Errors: 0
```



IBM
APPLE
QUME
NEC
SANYO
OTRONA
KAYPRO

APEX
Support Services



INSTALLATION
INSTRUCTION
REPAIR

Custom Cables
Downloading Services
Software Consulting
Maintenance Contracts
Preventative Maintenance

APEX INFORMATION SYSTEMS
2517 Van Ness Ave. (near Union)
San Francisco, Ca. 94109
Phone (415) 885-1633
24 Hour Pick-up & Delivery

Easy Pascal I/O Routines

Copyright 1982 by Michael A. Fischer

[Part One]

[Note: This article uses ALL CAPITALS to refer to procedures, functions and variables defined in this series of programs and "Quotes" to refer to predefined (intrinsic) procedures, functions and variables.]

Everyone knows that an infinite number of monkeys typing on an infinite number of word processors will eventually, after using an infinite number of reams of continuous feed paper, write the complete works of Shakespeare.

Any good programmer knows that a single gorilla, masquerading as a human typing in data to a computer program, can bomb all but the most carefully written program.

This series of articles presents numeric data input routines for UCSD Pascal which provides the means for creating programs which are "user-friendly". A user-friendly computer program requires, among other things, well-formatted screens, clear error messages, and "gorilla-proof" entry routines, which prevent the program from accepting improper values and keep the program from bombing if data is not entered in the proper format.

At first Pascal would not seem to be a good language for gorilla-proof programming; if string data is entered when the program is expecting a number the program terminates with an error message such as:

```
IO error:  bad input format
S# 1, P# 1, I# 15
Type <space> to continue.
```

and the system reinitializes when you press the space-bar.

But Pascal is an extendable language, with user-definable functions and procedures. And when you define your own data entry routines, you can include intelligible error messages rather than Basic's enigmatic, "REENTER". The ease of calling Pascal functions makes user-friendly programming easier in Pascal.

The first article in this series will present the theory behind the number input routines and will include the procedures and functions common to all the data input routines as well as the 4 functions for integer input.

Subsequent articles will discuss the real number (floating point) input routines, the long integer routines and a description of how to convert the routines to units so that they can be placed in your System Library rather than entered into each source code file that uses them and compiled with that file.

General Processing Routines

[The following discussion is based on listing one which follows this article.]

There are five general routines needed by all the other (i.e., integer, real or long integer) input routines.

The BLANKSTRIP procedure takes a string and strips off any leading or trailing blanks. It also returns a boolean variable set false if the string consists of all blanks. This routine uses the "Delete" procedure which deletes a specified number of characters beginning at a specified character in a string.

The SIGNPROC procedure removes any plus or minus sign in the string passed as a variable parameter and returns the string. If there was a minus sign it aslo sets the boolean variable MINUSSIGN to true. This can later be checked to determine if a positive or negative number was entered.

The SIGNPROC procedure uses the "Pos" function to determine the position of a plus or minus sign in the string. If the sign is in the first position it is deleted from the string and saved in the MINUSSIGN variable. If a plus or minus sign is found in any other posiition an error message is generated and an error flag is set. The flag can be checked by the calling routines and the message accessed and printed.

The STRIPZEROS procedure takes the string parameter and deletes leading zeros. It stops when it encounters a character other than '0' or when the length of the string is one.

The CLEARLINE procedure clears the screen from the current cursor position to the end of the current line. It does this by sending an "EOL" (ASCII character 29) to the monitor.

The real work of the general processing routines is done in the INTCONVERT procedure which converts a string to a long integer of a specified length and returns the long integer or an error flag and message if the conversion fails.

The INTCONVERT procedure begins by first calling BLANKSTRIP, SIGNPROC and STRIPZEROS. This results in a string without leading blanks or zeros, trailing blanks, or a leading plus or minus sign. The string is checked to see if it contains too many digits using the "Length" function. Then each character is checked to see if it is a digit and, if so, converted into a number. At any time if an error is detected the error flag is set and an appropriate error message is generated. Finally any minus sign is processed back into the number and the number is passed back to the calling routine as a long integer.

Integer Routines

The workhorse of the integer routines is the PROCINTEGER function. It takes a string entry, calls INTCONVERT and receives back a long integer with a length of five or less. It does this by calling INTCONVERT with the CHECKLENGTH (OKAYLENGTH) parameter set to five. Next it compares the long integer to "Maxint", a system defined constant of the largest permitted integer -- on the Apple this is 32767. It also compares the value to negative "Maxint". It finally converts the long integer to an integer using the "Trunc" function which changes a long integer to a regular integer. If conversion fails, it returns to the calling routine with the error flag set and an appropriate error message.

The four main integer functions are the ones which will actually be called by your program. The naming convention, which will also be used in the long integer and floating point routines, uses "i" (for integer) as the root and a suffix of "vert" (for convert). Finally each name may have a prefix of up to 2 letters -- a "P" if a prompt message is desired before the input, an "R" if the number must be within a specified range, a "Pr" if both options are wanted, and no prefix if neither option is selected.

Parameters passed to the functions consist of 4 integers -- an X and Y screen location for the prompt/data input, and an X and Y screen location for any error message. Additional parameters are required if the "P" option (the prompt line -- a string) or the "R" option (the low and high ends of the range as integers) are selected. The "P" option parameters preceed the "R" option parameters if both options are selected.

The routines are called by your program with a statement such as:

```
Input_variable := Pivert
(0,0,0,23,'Integer: ');
```

This would accept data beginning with a prompt message "Enter integer: " at screen location 0,0 (upper left corner) and print

MICRO Flash

COMPUTER SYSTEMS

ALL PRODUCTS ARE FACTORY-FRESH AND GUARANTEED!

PRODUCTS FOR APPLE & COMPATIBLES

HARDWARE & PERIPHERALS		
Advanced Logic CP/M Card	399.00	**299.95**
Hayes Micromodem II	379.00	**269.95**
Microsoft®		
Premium Softcard™ //e	495.00	**339.95**
Premium System™	695.00	**426.95**
Rana Elite One Disk Drive	379.00	**269.95**
Videx Ultraterm	379.95	**248.95**
Videoterm w/ Softswitch	319.00	**199.95**
Keyboard Enhancer II	149.00	**98.95**
SOFTWARE		
Ashton Tate dBase II	700.00	**398.95**
Friday!	295.00	**198.95**
BPI Systems Accounting	395.00	**309.95**
Broderbund Bank St. Writer	70.00	**52.95**
Continental Home Acct.	74.95	**45.95**
Microsoft® Multiplan™	275.00	**165.95**

MANY OTHER PRODUCTS AVAILABLE!

Please send certified or cashier's check to our **NEW ADDRESS:**

MICRO FLASH

1110 Burnett Avenue, Suite K
Concord, CA 94520

MasterCard VISA

Orders Only:
800/458-2433

For Price Quotes, Technical Service and California Orders, call:
415/680-1157

No Credit Card Surcharge
ONLY $6.00 UPS shipping/handling and insurance per order
Higher for printers & monitors—please call
CA residents add applicable sales tax
Items & prices subject to change
Minimum order $100.00
Dealer inquiries welcomed

any error message at screen location 0,23 (bottom left corner). When a proper integer has been entered its value will be placed in Input_variable (or whatever name you use).

Integer input error messages are:

"Misplaced + sign." -- The + sign used was not the first character.

"Misplaced - sign." -- The - sign used was not the first character.

"Non-number found." -- A non-number was found.

"Integer greater than +32767." -- The value of the integer is greater than the maximum value permitted for integers.

"Integer less than -32767." -- The value of the integer is less than the minimum value permitted for integers.

"Too many digits." -- The integer contained more than 5 digits.

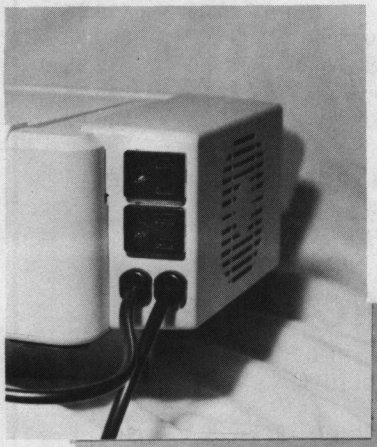
"Value too low; must be ### or more." -- The value entered was less than the minimum specified for the range.

"Value too high; must be ### or less." -- The value entered was greater than the maximum speicified for the range.

"No number entered." -- No number remains after removing leading blanks.

Each of the main functions begins by clearing the entry line from its starting position using the CLEARLINE procedure. Next the cursor is placed at the input location using the "Gotoxy" procedure, any prompt written, and data input accepted as a string. Next PROCINTEGER is called. If there is an error (OKAY=false), the error message line is cleared and the error message is written. If range checking is specified the range is checked and an error message written if the number (TEMP) is outside the specified range (less than LOW or greater than HIGH). Each routine repeats until there is no error (OKAY=true and, if range checking was specified, TEMP <= HIGH and >= LOW). When there is no error the function erases any error message and assigns the value of the entered number to the function.

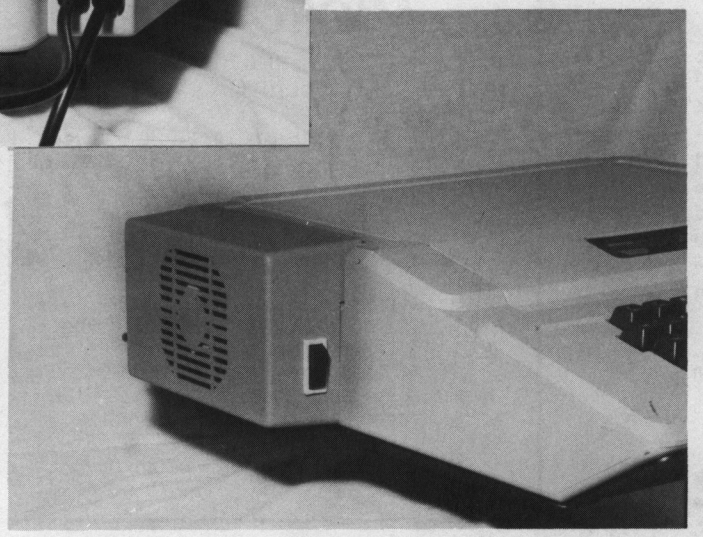
The main program in listing 1 is just a simple demonstration program using the entry routines. In actual use you could



Le Fan

pour la Pomme

-surge protection -matching colour
-illuminated switch -convenience outlets



$37.00 + tax

Orders may be placed and picked up at the General Meetings or by mail.

order from:
The San Francisco Apple Core, Inc.

1515 Sloat Blvd. Suite 2
San Francisco, CA

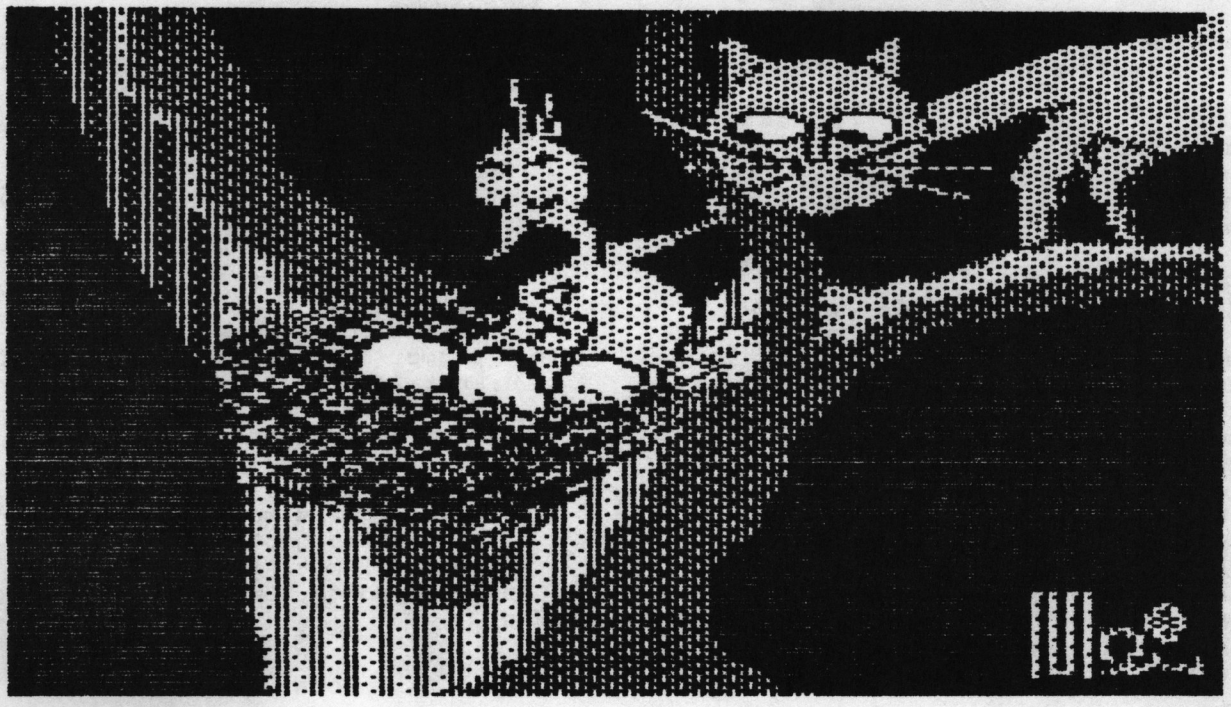
mail orders add $3.00 for shipping and handling.

replace it with your own main program and any of your own procedures and functions. Including the coding for all the procedures and functions adds considerable size to program text and code files. The use of the real (floating point) routines and the long integer routines, presented in later parts of this series, adds even more bulk. However a method will also be shown for installing these routines in your System.Library, making them callable by any program which includes the proper "Uses" statement in it.

```
(*$S+*)
Program Intiodemo;
(*This program demonstrates the integer
input routines.*)
Type
 Long = Integer[36];
Var
 Ch                    : Char;
 Highirange,
 Intvalue,
 Lowirange,
 Selection,
 Xerr,
 Xloc,
 Yerr,
 Yloc,
 Ypos                  : Integer;
 Promptline            : String;

Procedure Blankstrip (Var Entry:String;
Var Okay:Boolean);
(*This routine removes leading and
trailing blanks, checking that non-blank
characters remain.*)
 Procedure Checklength
(*This routines checks that the legnth of
the entry is not zero.*)
  Begin
   If (Length(Entry) = 0)
    Then
     Begin
      Okay := False;
      Exit (Blankstrip)
     End
  End;

 Begin (*Blankstrip*)
  Checklength;
  While (Entry[1] = ' ') Do
   Begin
    Delete (Entry,1,1);
    Checklength
   End;
  While (Entry[Legnth(Entry)] = ' ') Do
   Begin
    Delete (Entry,Length(Entry),1);
    Checklength
   End;
  Okay := True
 End;

Procedure Signproc (Var Entry:String; Var
Minussign:Boolean; Var Okay:Boolean; Var
Errmessage:String);
(*This routine removes any + or - sign,
checks if it is in the correct location
and saves the minus sign for later use.*)
Var
 Minus,
 Plus                  : Integer;
Begin
 Plus := Pos ('+',Entry);
 Minus := Pos ('-',Entry);
 If (Plus > 1)
  Then
   Begin
    Errmessage := 'Misplaced + sign.';
    Okay := False;
    Exit (Signproc)
   End;
  If (Minus > 1)
   Then
    Begin
     Errmessage := 'Misplaced - sign.';
     Okay := False;
     Exit (Signproc)
    End;
   If (Plus = 1) or (Minus = 1)
    Then Delete (Entry,1,1);
   Minussign := (Minus = 1);
   Okay := True
 End;

Procedure Stripzeros (Var Entry:String);
(*This routine removes leading zeros from
the number.*)
 Begin
  While (Entry[1] = '0') and
        (Length(Entry) > 1) Do
   Delete (Entry,1,1)
 End;

Procedure Clearline (X,Y:Integer);
(*This routine clears the screen from the
cursor position to the end of the line.*)
 Begin
  Gotoxy (X,Y);
  Write (Chr(29))
 End;

Procedure Errange (Procname:String);
(*This procedure ends the program and
prints the error message if the low value
passed to the range checker is higher than
the high value.*)
 Begin
  Page (Output);
  Gotoxy (0,10);
  Writeln ('Range error in', Procname,
    '; low');

  Writeln ('value greater than high',
    'value.');
  Writeln ('Program will end.  Please',
    'correct your');
  Writeln ('program, recompile and ',
    're-run.');
  Exit (Program)
 End;

Procedure Intconvert (Entry:String;
Okaylength:Integer; Var Value:Long; Var
Okay:Boolean; Var Errmessage:String);
(*This routine converts a string to a long
integer and reports any errors with
appropriate error messages.*)
```

```
Var
 Minussign                : Boolean;
 Counter                  : Integer;
Begin
 Okay := True;
(*Remove blanks.*)
 Blankstrip (Entry, Okay);
(*If nothing left - error.*)
 If (Not Okay)
  Then
   Begin
    Errmessage := 'No number entered';
    Exit (Intconvert)
   End;
 Value := 0;
(*Remove any sign.*)
 Signproc (Entry, Minussign, Okay,
   Errmessage);
 If (Okay)
  Then
   Begin
(*Remove zeros.*)
    Stripzeros (Entry);
(*See if too many digits in number.*)
    If (Length(Entry) > Okaylength)
     Then
      Begin
       Errmessage := 'Too many digits.';
       Okay := False;
       Exit (Intconvert)
      End;
(*Go through each character left.*)
    For Counter := 1 to Lenght (Entry) Do
     If Not (Entry[Counter] in
        ['0'..'9'])
      Then
(*If non-number, exit with error
message.*)
       Begin
       Errmessage:= 'Non-number found.';
       Okay := False;
       Exit (Intconvert)
       End
(*If numeric, convert to digit after
increasing number so far to show moved one
place to the left.*)
      Else Value := Value * 10 + Ord
(Entry[Counter]) - Ord ('0')
   End;
(*Make negative if minus sign present.*)
 If (Minussign)
  Then Value := -Value
End;

Function Procinteger (Entry:String; Var
Okay:Boolean; Var Errmessage:String) :
Integer;
(*This routine takes the string entry,
sets it maximum length to five digits,
converts it to a long integer, checks
whether it exceeds maxint, and converts it
to an integer.*)
 Const
  Okaylength = 5;
 Var
  Longone             : Long;
 Begin
  Intconvert (Entry, Okaylength, Longone,
    Okay, Errmessage);
  If Not (Okay)
   Then Exit (Procinteger);
  If (Longone > Maxint)
   Then
    Begin
     Errmessage := Concat ('Integer',
'greater than + 32767.');
     Okay := False;
     Exit (Procinteger)
    End;
  If (Longone < -Maxint)
   Then
    Begin
     Errmessage := Concat ('Integer',
'less than -32767.');
     Okay := False;
     Exit (Procinteger)
    End;
  Procinteger := Trunc (Longone)
End;

(*The following routines accept string
input and return either an error message
or the value of the string as an integer.
Each accepts input or prints a prompt at a
specificed location and prints error
messages at another specified location
(passed as the first four integer
parameters).  Function names consist of
the root i (for integer), the suffix
"vert" (for convert), and zero to two
letters as prefixes.  The prefixes are "p"
(for prompt - pass prompt string as
parameter), "r" (for range - pass low and
high values of range as parameters), and
"pr" (for both - pass prompt parameter
before range parameter.*)

Function Ivert (X,Y,Errx,Erry:Integer) :
Integer;
 Var
  Okay                : Boolean;
  Temp                : Integer;
  Entry,
  Errmessage          : String;
 Begin
  Repeat
   Clearline (X,Y);
   Gotoxy (X,Y);
(*Get entry.*)
   Readln (Entry);
(*Convert it.*)
   Temp := Procinteger (Entry, Okay,
Errmessage);
   If Not (Okay)
    Then
(*Print error message.*)
     Begin
      Clearline (Errx,Erry);
      Gotoxy (Errx,Erry);
      Write (Errmessage)
     End
  Until (Okay);
(*Erase error message line.*)
  Clearline (Errx,Erry);
  Ivert := Temp
 End;

Function Pivert (X,Y,Errx,Erry:Integer;
Prompt:String) : Integer;
 Var
  Okay                : Boolean;
  Temp                : Integer;
```

```
  Entry,
  Errmessage          : String;
 Begin
  Repeat
   Clearline (X,Y);
   Gotoxy (X,Y);
(*Prompt for entry.*)
   Write (Prompt);
   Readln (Entry);
(*Convert it.*)
   Temp := Procinteger (Entry, Okay,
Errmessage);
   If Not (Okay)
    Then
(*Print error message.*)
     Begin
      Clearline (Errx,Erry);
      Gotoxy (Errx,Erry);
      Write (Errmessage)
     End
  Until (Okay);
(*Erase error message line.*)
  Clearline (Errx,Erry);
  Ivert := Temp
 End;

Function Rivert (X, Y, Errx, Erry, Low,
High:Integer) : Integer;
 Var
  Okay                : Boolean;
  Temp                : Integer;
  Entry,
  Errmessage          : String;
 Begin
  If Low > High
   Then Errrange ('Function Rivert');
 Repeat
   Clearline (X,Y);
   Gotoxy (X,Y);
(*Get entry.*)
   Readln (Entry);
(*Convert it.*)
   Temp := Procinteger (Entry, Okay,
Errmessage);
   If Not (Okay)
    Then
(*Print error message.*)
     Begin
      Clearline (Errx,Erry);
      Gotoxy (Errx,Erry);
      Write (Errmessage)
     End
(*Check range limits.*)
    If (Okay) and (Temp < Low)
     Then
      Begin
       Clearline (Errx,Erry);
       Gotoxy (Errx,Erry);
       Write ('Value too low; must be',
Low, 'or more.')
      End;
     If (Okay) and Temp > High)
      Then
       Begin
        Clearline (Errx,Erry);
        Gotoxy (Errx,Erry);
        Write ('Value too high; must be',
High, 'or less.')
       End
  Until (Okay) and (Temp <= High) and
(Temp >= Low);
  Clearline (Errx,Erry);
  Rivert := Temp
 End;

Function Privert (X, Y, Errx, Erry :
Integer; Prompt : String; Low,
High:Integer) : Integer;
```



```
 Var
  Okay                : Boolean;
  Temp                : Integer;
  Entry,
  Errmessage          : String;
 Begin
  If Low > High
   Then Errrange ('Function Privert');
 Repeat
   Clearline (X,Y);
   Gotoxy (X,Y);
(*Prompt for entry.*)
   Write (Prompt);
   Readln (Entry);
(*Convert it.*)
   Temp := Procinteger (Entry, Okay,
Errmessage);
   If Not (Okay)
    Then
(*Print error message.*)
     Begin
      Clearline (Errx,Erry);
      Gotoxy (Errx,Erry);
      Write (Errmessage)
     End
(*Check range limits.*)
    If (Okay) and (Temp < Low)
     Then
      Begin
       Clearline (Errx,Erry);
       Gotoxy (Errx,Erry);
       Write ('Value too low; must be',
Low, 'or more.')
      End;
     If (Okay) and Temp > High)
      Then
       Begin
        Clearline (Errx,Erry);
        Gotoxy (Errx,Erry);
        Write ('Value too high; must be',

High, 'or less.')
       End
  Until (Okay) and (Temp <= High) and
(Temp >= Low);
  Clearline (Errx,Erry);
  Privert := Temp
 End;

Procedure Intro;
(*Put introductory explanation on
screen.*)
 Begin
  Page (Output);
  Gotoxy (3,0);
  Writeln ('Numeric Input Conversion',
      'Routines');
  Gotoxy (9,1);
  Writeln ('Demonstration Program');
  Writeln;
  Writeln ('This program demonstrates ',
    'the use of ');
  Writeln ('string inputs converted to ',
    'integers,');
  Writeln ('avoiding run time errors ',
    'resulting from ');
  Writeln ('entering non-numeric data ',
    'when numeric');
  Writeln ('data is expected.');
  Writeln
 End;
```

```
Procedure Setup;
(*This procedure gets the various data
about the routine to be shown.  This
includes whether range checking is
desired, and whether a prompt line should
be displayed.*)
 Begin
(*Selection is key to selection of proper
numeric input routine.*)
  Selection := 0;
(*Get location for data or prompt input.*)
  Xloc := Privert (0, 10, 0, 23,
    'Enter X position for input: ', 0,79);
  Yloc := Privert (0, 11, 0, 23,
    'Enter Y position for input: ', 0,23);
(*Get location for error message.*)
  Xerr := Privert (0, 12, 0, 23,
   'Enter X position for error message: ',
   0, 79);
  Yerr := Privert (0, 13, 0, 23,
   'Enter Y position for error message: ',
   0, 23);
(*Determine if range checking option
wanted.*)
 Repeat
  Gotoxy (0,14);
  Writeln ('Do you want range checking ',
    '(Y/N): ');
  Read (Keyboard, Ch)
 Until (Ch in ['Y','y','N','n']);
 Ypos := 15;
 If (Ch='Y') or (Ch='y')
  Then
   Begin
    Selection := Selection + 2;
(*If range checking, get high and low
range values.*)
    Lowirange := Pivert (0, Ypos, 0, 23,
      'Enter low value: ');
    Ypos := Ypos + 1;
    Highirange := Pivert (0, Ypos, 0, 23,
      'Enter high value: ');
    Ypos := Ypos + 1;
   End;
(*Determine if prompt option wanted.*)
 Repeat
  Gotoxy (0, Ypos);
  Writeln ('Do you want a prompt line ',
    '(Y/N): ');
  Read (Keyboard, Ch)
 Until (Ch in ['Y', 'y', 'N', 'n']);
 Ypos := Ypos + 1
 If (Ch = 'Y') or (Ch = 'y')
  Then
   Begin
    Selection := Selection + 1;
(*If prompt option, get prompt line.*)
    Write ('Prompt text: ');
    Readln (Promptline);
    Ypos := Ypos + 1
   End;
 Gotoxy (0, Ypos);
 Writeln ('Press space bar to clear ',
   'screen and get');
 Writeln ('input.')
 Read (Keyboard, Ch)
End;
```

CONT. PG. 27

Marketing

by

Don Scellato

The mainstay of our Sales Table was absent from his voluntary position at the last meeting because he was having surgery. Joe Bianco will be back at his post in time for the Applefest and will be back at the table for the next meeting. I hope all of the readers join me in saying thanks for the many hours Joe has put in at the table and in wishing him a successful surgery and speedy recovery.

We've had many demands for a new Library to join Library 1, Library 2, and the 1981 Yearbook. Due to a transition in the Librarian position and in the Sales Director position, there were not enough disks in 1982 and in the beginning of 1983 to make a meaningful Library of the disks of the month from those years.

I plan now to place the disks of the month from 1982 and the collection of 1983 disks (which is now up to three and will hopefully be up to six by the end of the year) in a Library 3 (1982-1983) which will be available in the Clubhouse for copying in late January and will be available for sale in February.

The three Libraries of disks that are in the Clubhouse have recently been restored to mint condition and the rather dogeared libraries that were there for the past year have been destroyed.

The VisiCalc group is still working on SFAC VisiCalc Template Disk # 2. This disk will contain a collection of tax preparation templates based on 1982 law that may be adapted by the user to 1983 tax law as soon as the tax forms are formally revised and published. It will also contain an attendence template for school teachers and a class grading template.

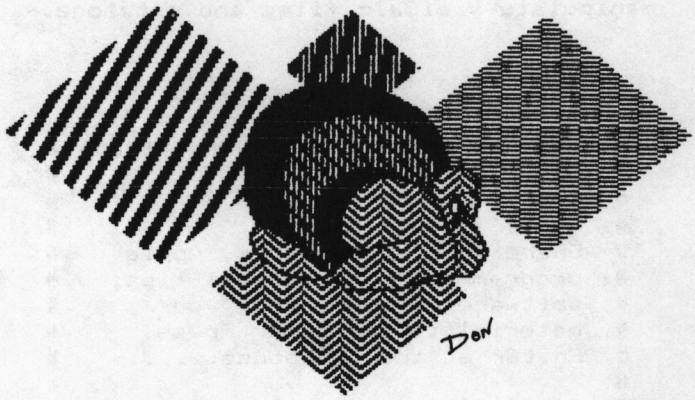
The preparation of these templates takes quite a bit of time. They're shooting for release at the Applefest, but might not make it till the November meeting. We're always looking for programs and for spreadsheet templates for inclusion on SFAC disk releases.

Just recently we received a Lo-Res graphics and text adventure game called Escape Into Espionage from Daniel Lipkin of Oakland. This game will be included on our next general purpose disk release (SFAC 1983 Disk No. 4) along with some programs contributed by Andrew D. Falenski of Seaside, Oregon. Both Daniel and Andrew may request a free copy of any of the Club's disks by calling the Clubhouse or by writing to me care of the Clubhouse.

We will be releasing Blossoming Forth, a Word Processor and Forth Tutorial, at the Applefest. If you can't pick it up there, you can get a copy via the order blank in the back of the Cider Press or at the next General Meeting. This software was entered in the Programming Contest by Dudley Ackerman and Ed Bronstein of the Forth Interest Group. It should be an asset for anyone interested in learning Forth and in working with the language.

Disk Drive and controller card sales have been brisk. We will continue to offer these items along with cooling fans to members at the prices advertised in this issue of the Cider Press. Due to an increase in the cost of fans to the Club, fans will be sold for $37.50 plus tax at all future meetings. We will continue the sale of blank diskettes for $20 for a box of 10 diskettes. Shift key mods are now available at the Sales Table for $7.50.

A number of disk drive orders were placed by members at the last meeting. These orders will be filled at the next meeting. If you need another disk drive and like the price, contact me at least 10 days before the November meeting and I'll add your name to the list.



Interest Groups

VisiCalc

by

Don Scellato

The San Francisco Visicalc Interest Group meeting for October was cancelled because the group leader was stuck in Los Angeles on a business trip. The East Bay Visicalc Interest Group met at Breuner's in Pleasant Hill at 7 P.M. on Monday, October 17. The attendees of this meeting continued discussions of tax applications for VisiCalc.

Then the group leader demonstrated each of the VisiCalc commands for the benefit of the newer users who attended the meeting. A number of templates were exchanged by the attendees in a continuing effort to help one another to master the program.

The next meeting of the San Francisco contingent of the VisiCalc Interest Group will be held at 7:30 P.M. at the Clubhouse on Tuesday, November 8. Anyone who is planning to attend should try to vote in the morning.

We will continue discussions of tax preparation techniques. Beginners are welcome at these meetings. You don't have to be an expert. We'll take the time to demonstrate anything you might want to see.

The next meeting of the East Bay VisiCalc Interest Group will be held at Breuner's on Buskirk Avenue in Pleasant Hill in the upstairs meeting room from 7 P.M. till 9 P.M. on Monday, November 21.

Discussions of tax preparation will continue. We will also demonstrate and discuss a number of programs which manipulate VisiCalc files and catalogs.

```
########################################
#                                      #
#          AUTHORS WANTED              #
#                                      #
#  Share your experiences, Apple       #
#  programming hints,pascal tips,      #
#  software reviews. Send your         #
#  material to the Cider Press         #
#  Editor at the Clubhouse.....        #
#                                      #
########################################
```

Telecommunications

****
****

The communications group is one of the more recently formed interest groups within the San Francisco Apple Core. We held our first meeting at the end of May & have held meetings on the fourth Wednesday of every month since then. (note we have changed from the above schedule for November and December because other wise our meetings would clash Thanksgiving & Christmas).

Like most other interest groups our meetings are attended by people with a broad spectrum of knowledge & experience. We usually have a demonstration of some communications product for the Apple - (hardware or software). We then discuss problems we have encountered etc.

You may by now be asking what is Computer communications ?. Why should I care about it ? There are now millions of computers in use every day all over the world. There are many ways of transmitting information from one computer to another. The most common method is to send & receive data over the telephone using a device known as a Modem & some special communications software.

Using these items users of home computers can send programs or "electronic mail" to friends . They can call Bulletin Boards - where they can leave & recieve messagess. They can work at home & communicate with the office using the modem. They can send Electronic mail - MCI mail & E-com are examples of such services. Electronic shopping & banking can also be carried out using your Apple & a modem.

If you are interested in using your Apple for Electronic communications but are totally confused by the terminology and the large range of hardware and software available, we invite you to attend our November meeting which will be held at the club house on November 9th.

Group leader

Frank Fennelly
940 I Kiely
Blvd.
Santa Clara,
Ca.95051

(408) 984-7321

Kids Can't Wait

The Apple Core's participation in Apple Computer's "Kids Can't Wait" computer giveaway program is now under way. We are now accepting volunteers for the program. Volunteers will act as consultants to schools needing more help than the computer stores can offer.

Computer stores involved in the program have provided only four hours of training to schools.

If you called before regarding volunteering for the program please call again. There is a coordinator in each of the four counties that we have been assigned to assist. If you have some time or have a school you want to help, please contact the coordinator in that area.

The coordinators are as follows:

Marin County
Reed Harris
924-7221

Contra Costa
Allan B. Woods
687-9659

San Francisco
Ben Lai
564-7773

Alameda
Joel Brodsky
655-1255

EXTRAVAGANZA

The next general meeting will be hosted by the interest groups. If you want to know what actually goes on at any of the group meetings, come to the Interest Group Extravaganza Nov. 19th at Fort Mason.

If you are interested in groups other than those always listed in the Cider Press, then this will be a time to find others with like interests. If you want an announcement in terms of a new interest or education type group, contact me (Sami Capron) at 388-8631 prior to the meeting so that I can prepare an announcement for you.

Hope to see you all at the Extravaganza!

CP/M

by Jim Ayers

The CP/M users are fat and sassy and will continue meeting at 301 Poplar St in Mill Valley. This is not a long way from San Francisco, it is only over the bridge (the Golden Gate that is) and thru the Rainbow tunnel also known as the Waldo. Then when you get to the turn-off that is known as the Stinson Beach exit, or Highway 1, about 1.5 miles from the tunnel, you take that road. You then go passed Howard Johnson Motel and Resturant to the stop light, turn left, travel 1/2 mile the block passed the 7-11 store, and there you will see a 'state of the art' sign, a traveling LED and below that sign, you will find the small and we hope intelligent, active and interested CP/M users group.

There is a nucleus of people that can help you learn, here at the CP/M group. Don't come and expect to be handed knowledge, it doesn't come that way, but if you come expecting the best, that is what you will get. There is, under that roof, a library of knowledge and programs beyond your kin, that you may have access to, if you will but come. The second and fourth Tuesdays, at 7:30 PM. We will hope to see you there. You may also call the RBBS of Marin, with 8 Mega Bytes of programs on line, to download to YOUR very OWN system. The magic number is (415)-383-0473 from 5:00 PM to 8:30 AM Monday thru Thursday and 5:00 PM Friday evening to 8:30 AM the following Monday morning. Come one come all,

2nd and 4th Tuesdays, 7:30 PM
301 Poplar St, Mill Valley, CA
See you there!

PASCAL - CONT.

```
Begin (*Main*)
 Repeat
  Intro;
  Setup;
  Page (Output);
  Case Selection of
   0 : Intvalue := Ivert (Xloc, Yloc,
Xerr, Yerr);
   1 : Intvalue := Pivert (Xloc, Yloc,
Xerr, Yerr, Promptline);
   2 : Intvalue := Rivert (Xloc, Yloc,
Xerr, Yerr, Lowirange, Highirange);
   3 : Intvalue := Privert (Xloc, Yloc,
Xerr, Yerr, Promptline, Lowirange,
Highirange);
  End; (* Case *)
  Write ('The value entered was ');
  Writeln (Intvalue);
  Repeat
   Gotoxy. (0,23)
   Writeln ('Another number (Y/N)?');
   Read (Keyboard, Ch)
  Until (Ch in ['Y', 'y', 'N', 'n'])
 Until (Ch = 'N') or (Ch = 'n')
End.
```

Motor's Manual For Apple

A Book Review by Joyce Conklin
(c) 1982

HOW TO REPAIR THE APPLE COMPUTER
(c) 1982 by DataLine
2961 S. Las Palmas, Mesa, AZ 85202
List Price: $39.95

This small manual (listed as Vol. 1 because there are others in the planning stage) is printed in eye-ease (and xerographic-defeating) blue. When you see the size and compare it to the price, you might think it's another rip-off such as was more common in the earlier days of microcomputers as everyone tried to get on the bandwagon. But NOT SO... as is pointed out, if you have a reasonable degree of mechanical ability, one fix will save you the price of a service call and then some.

There is a disclaimer of responsibility on the part of the author -- all boo-boo's are your fault. But if you were really worried about that, you wouldn't try to fiddle with the computer anyway.

With that aside, you start reading the book and find that it really has the "ordinary" user in mind, not the electronic hobbyist. There is a diagram of an IC chip with arrows to show how to find the reference point (usually called Pin 1); this is necessary to know if you are going to remove and replace chips. The pins are all evenly spaced so it is possible to get the chip in backwards with unpredictable results.

The description of the various parts is in layman's language and the directions are very nicely given in a step-wise manner so that you can follow them. It's a good idea to read through the entire book first (after all, how long can it take to read 56 pages?) and then do what is advised. I discovered that the way I installed my TKC keypad was a no-no! Guess I was just lucky.

You are advised never to do repairs on the carpet, even if that is a nice soft place and gives a large area to place the unattached parts. The carpet can hold static electricity which is anathema to IC's. Like I say, I was lucky.
The directions include "GOTO" statements which allow for variations in the type of computer which you may have - this book will work for Apple][, as well as Apple][+ in it's many revisions.

The chapters include Trobule-shooting Overview, Problems with the Motherboard, Problems with the Power supply, Problems with the Keyboard, and Final Notes. In each case you are led through the process of trouble-shooting one step at a time. As I was reading through the book, I decided to look at the inside of my Apple and noticed for the first time the amount of dust mixed with grease that had accumulated on the motherboard from my fan. A careful dusting with clean short-bristled brush (actually an unused oil-painting brush), a swoosh with the Dust-Buster, and I was back in business.

Who knows when I would have noticed this problem, if I had not had the cover off while looking at each of the parts mentioned. The trouble-shooting chart looks like the ones you picked up to repair your TV tubes (you don't remember tubes? my oh my, are you young!) and shows which chips may be at fault for different problems. This is after you have checked out other potential problems such as bent pins (you shouldn't have any if you haven't been fooling around with the chips) or a board in the wrong slot (it's really fun trying to PR#6 when your controller card is in slot #7!).

Listed in the appendix are some mail-order electronic supply houses, so if you don't have an electronic shop in your town, you can still obtain the supplies.
If you aren't planning any repairs at the moment but just want to add on some of the many gadgets which are supplied for the Apple, this book is worth it in teaching you EXACTLY how to take the case apart (incidentally, the author claimed never to have run into ANY problems with the computer which were caused by the case!) and put it back

together again, without removing vital parts - the screws that hold the motherboard in place, for example! How are you going to know not to unscrew those four at the back of the case bottom? They look like all the rest from the outside.

All in all, this is a very easy-to-read-and-follow book which may come in handy, and when it does, you will bless the day you had the foresight to invest in this rather inexpensive insurance policy. It's a real shock when you boot your favorite applications program and it doesn't work, and neither does the back up disk.

It's a real relief (no, that's not spelled R-O-L----- in this case) when you remember what you read and go inside to make sure all the connections are tight; a little pressing on the cards does the trick and your 64K program will now load. I recommend this book to all but the complete idiot, which by definition doesn't include any readers of this publication!

= + = + = + = + = + = + = + = + =

Bus Service and Directions to Fort Mason Center

FortMason can be reached by the following Muni lines.

#22 — Fillmore (Fillmore and Laguna)

#28 — 19th Ave. (terminates inside west gate of FMC)

#30 — Stockton (Laguna and Chestnut)

#42 — Downtown Loop (VanNess and Bay)

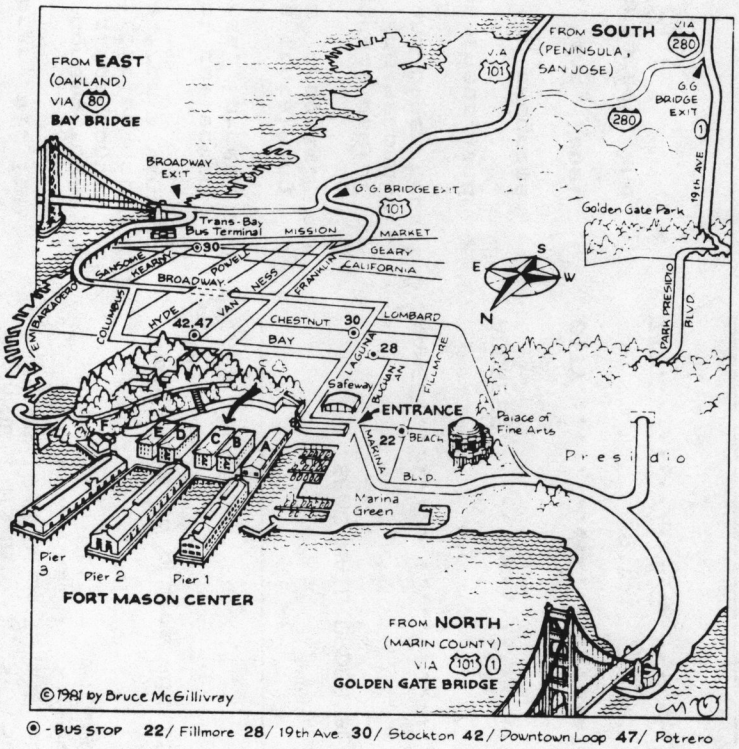
#47 — VanNess (VanNess and Bay)

These are the bus stops closest to FMC. For further information call 673-MUNI.

Golden Gate Transit buses intersect #28-19th Ave. buses at the S.F. G.G. Bridge toll plaza. Golden Gate Transit — 332-6600.

DIRECTIONS: **EAST**—Bay Bridge to Broadway Exit—Broadway to Laguna, Rt. turn to Marina Blvd. & Buchanan St.—Rt. turn. **Peninsula**—U.S. 101 North to Franklin St. Exit. Franklin to Bay, lft. turn to Laguna, Rt. turn to Marina Blvd. Rt. turn at Buchanan.

Pedestrian entrance at Laguna. Auto entrance at Buchanan.



FOR MEETING INFORMATION CALL (415) 566-2342

APPLE CORE OFFERINGS

NOTES

SFAC 1983 DISK NUMBER 1 contains only three programs due to the nature of one of the programs - a hi-res graphics game. This game is called "The Incredible Shrinking Man" which was contributed by the Chiang Brothers. Also included on the disk is a High Score Tabulator and hi-res graphics display program called Gallery.

LIBRARY #1 contains over 340 programs. It is packaged in diskette holders and is bound in a SF Apple Core binder with over 30 pages of documentation. This documentation alone is available for three dollars.

LIBRARY #2 continues the tradition started with Library #1. It contains over 250 programs from the 1980 DOMs.

1981 YEARBOOK contains 12 disks containing all DOM programs from 1981. There is a heavy emphasis on games and business applications.

1983 PROGRAMMING UTILITIES contains a collection of utilities not collected on a single disk before. These utilities come from a variety ofsources such as IAC disks, 1981 DOMs, and 1982 DOMs.

1983 GAME SET consists of two disks containing games not collected in a single set before. It contains a variety of hi-res, lo-res, and text games.

1983 EDUCATION APPLICATIONS is a set of two disks containing interactive learning programs, a grade maintenance program, and GRAMMAR QUEST an interactive learning game.

1983 HOME APPLICATIONS consists of two disks of programs useful around the home. One disk was from the 1982 Home Pak. The second disk is a collection of programs gathered from IAC disks and other recent contributions.

SFAC 1983 DISK 2 is a collection of game, education, business, and utility applications.

SFAC 1983 DISK 3 is a collection of useful and fun programs including a HI-RES/LO-RES Baseball Game.

SFAC MAILING LIST is an inexpensive mailing list and label printing program.

SFAC PASCAL DISK 1 is a collection of Pascal programs and utilities contributed by Max Nareff.

PRODUCT	MTG PRICE	MAIL PRICE
Library #1	$98.00	$101.50
Library #2	$98.00	$101.50
1981 Yearbook	$98.00	$101.50
COLLECTIONS		
Basic Tut Pak(3)	$20.00	$22.00
Forth Pak (3)	$20.00	$22.00
Game Set (2)	$15.00	$17.00
Home Apps(2)	$15.00	$17.00
Education App(2)	$15.00	$17.00
1983 Prog Util(1)	$10.00	$12.00
VisiCalc Temp1(1)	$10.00	$12.00
VisiCalc Util (1)	$18.00	$20.00
IAC BASE	$10.00	$12.00
Mailing List	$10.00	$12.00
SFAC Disk 1	$ 7.50	$ 9.50
SFAC 1983 Disk 2	$ 7.50	$ 9.50
SFAC 1983 Disk 3	$ 7.50	$ 9.50
SFAC 1983 Pasc 1	$ 7.50	$ 9.50
Mach Lang Util	$10.00	$12.00

All foreign orders add 25% to above prices.

SFAC MAIL ORDER BLANK

NAME.....

ADDRESS.....

CITY.....STATE.....

ZIP.....

November, 1983

Monday	Tuesday	Wednesday	Thursday
	1 East Bay 7:30 Forth 7:30		3 Green Apples 8:00
7 Pascal 7:30	8 CP/M 7:30 VisiCalc 7:30	9 Marin Graphics 7:30 Tel'comm 7:30 Board Mtg.7:00	
	15 Forth 7:30	16 Apple /// 7:30 Nu'ers 7:30	17 VisiCalc East Bay 7:00
21 Pascal 7:30	22 CP/M 7:30 Epson 7:30		

December, 1983

Monday	Tuesday	Wednesday	Thursday
			1 Green Apples 8:00
	6 East Bay 7:30 Forth 7:30	7 Board Mtg. 7:00	
12 Pascal 7:30	13 CP/M 7:30 VisiCalc 7:30	14 Marin Graphics 7:30 Tel'comm 7:30	
	20 Forth 7:30	21 Apple /// 7:30 Nu'ers 7:30	22 VisiCalc East Bay 7:00 Green Apples 8:00
26 Pascal 7:30	27 CP/M 7:30 Epson 7:30		29 Green Apples 8:00

Check with Group Leaders About Holiday Season Meetings.

INTEREST GROUP LEADERS

Apple ///	Don Norris	673-7635 (day)
CP/M	Jim Ayers	383-0473(day)
		383-5759(eves)
East Bay	Molly Mitchell	526-9541
Epson	Michael Fischer	557-1250 (day)
		933-4435(eve)
Forth	Dudley Ackerman	626-6295
Green Apples (BASIC)	Gary Owens	521-4002
Marin Graphics	Reed Harris	924-7221
	Bob Sachs	332-3331
Nu'ers	Cathy Rudovsky	756-6480
Pascal	Mike Schnapp	386-2238
Product Review	"Woody" Liswood	585-6375
Telecommunications	Frank Fennelly	(408)984-7321
VisiCalc	Don Scellato	672-2051(eve)

San Francisco Apple Core, Inc.

1515 Sloat Blvd., Suite 2, San Francisco, CA 94132

(415) 566-2342

The San Francisco Apple Core is a non-profit corporation whose purpose is to share information about Apple computers through member services, interaction and the club newsletter. The SFAC is run by volunteer officers appointed by an annually elected Board of Directors. Membership in the SFAC is not limited to Apple owners.

GENERAL MEETINGS are held monthly in Bldg. C, Fort Mason Center, San Francisco, CA (see map on reverse side) Meetings include demonstrations of products and techniques and are scheduled at 10:00 a.m. on the third Saturday of the month. (Occasional scheduling conflicts may occur - phone **clubhouse** for changes. (415) 566-2342) "Interest groups" meet at various locations in the S.F. Bay Area and focus on specific areas of Apple use i.e. Visicalc, Basic, Pascal, etc.

The **CIDER PRESS** is the official publication of the SFAC, published approximately 10-12 times a year, mailed bulk rate to members in the United States, and air mail to members in foreign countries (Canada - first class). The **CIDER PRESS** includes club news, program listings, tips, reviews and special features which are CONTRIBUTED BY MEMBERS.

PROGRAM DISKS are assembled from **PUBLIC DOMAIN, NON-COPYRIGHTED** programs donated by members. These disks may contain demonstration programs, utilities, games, educational programs and are available for purchase by members at the general meetings or by mail. Collections of these disks called LIBRARIES are also available from the club. Details regarding purchases of these and other items may be found in the CIDER PRESS.

TO BECOME A MEMBER, complete and return the application form below with the appropriate payment (U.S. Funds).

All applications must be in the name of a **person**. Company or organization names are not accepted.

The amounts listed below include the $20.00 annual dues plus foreign mailing costs where applicable.

USA	$20.00	South America, Europe, North Africa	$41.00
Canada	$30.00	Eastern Europe, Asia, Pacific Isles	$47.00
Mexico, Carribean, Central America	$30.00	Middle East, Australia, South Africa	$47.00

Detach along dotted line

Application for Membership

SAN FRANCISCO APPLE CORE

FOR OFFICE USE ONLY		

Name _____

Address _____

City _____ State _____ Postal Code _____

USA ☐ Other _____

Phone Number (Home) _____ (Work) _____

Check here if you have been a member previously ☐

Send application form and payment to:

S.F. APPLE CORE
1515 Sloat Blvd., Suite 2
San Francisco, CA 94132

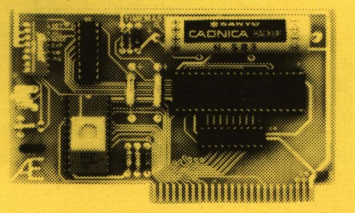
Amount Enclosed $_____

Apple Peripherals Are All We Make

That's Why We're So Good At It!

NEW

The TIMEMASTER
Finally, a clock that does it ALL!



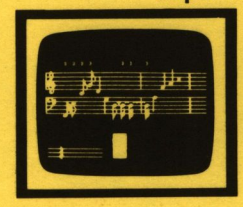
- Designed in 1983 using I.C. technologies that simply did not exist when most other Apple clocks were designed.
- Just plug it in and your programs can read the year, month, date, day, and time — down to 1 millisecond!
- Powerful 2K ROM driver — No clock could be easier to use.
- Full emulation of most other clocks, including Mountain Hardware's Appleclock (but you'll like the TIMEMASTER mode better).
- Compatible with all of Apple's languages, CP/M and PASCAL software on disk.
- Eight software controlled interrupts so you can execute two programs at the same time. (Many examples are included)
- On board timer lets you time any interval up to 48 days long down to the nearest millisecond.

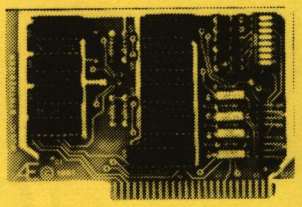
The TIMEMASTER includes a disk with some really fantastic time oriented programs (over 25) plus a DOS dater so it will automatically add the date when disk files are created or modified. This disk is over a $200.00 value alone — we give the software others sell. All software packages for business, data base management and communications are made to read the TIMEMASTER.

If you want the most powerful and the easiest to use clock for your Apple, you want a TIMEMASTER.

PRICE $129.00

Super Music Synthesizer

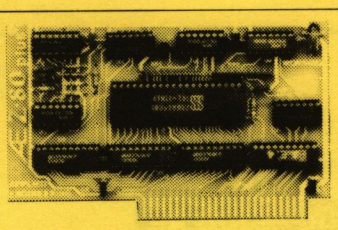




- Complete 16 voice music synthesizer on one card. Just plug it into your Apple, connect the audio cable (supplied) to your stereo, boot the disk supplied and you are ready to input and play songs.
- It's easy to program music with our compose software. You will start right away at inputting your favorite songs. The Hi-Res screen shows what you have entered in standard sheet music format.
- Now with new improved software for the easiest and fastest music input system available anywhere.
- We give you lots of software. In addition to Compose and Play programs, the disk is filled with over 30 songs ready to play.
- Easy to program in Basic to generate complex sound effects. Now your games can have explosions, phaser zaps, train whistles, death cries. You name it, this card can do it.
- Four white noise generators which are great for sound effects.
- Plays music in true stereo as well as true discrete quadraphonic.
- Full control of attack, volume, decay, sustain and release.
- Will play songs written for ALF synthesizer (ALF software will not take advantage of all the features of this board. Their software sounds the same in our synthesizer.)
- Automatic shutoff on power-up or if reset is pushed.
- Many many more features.

PRICE $159.00

NEW



Z-80 PLUS!

- An on-card PROM eliminates many I.C.'s for a cooler, less power consuming board. (We use the Z-80A at a fast 3.58 MHZ)
- Does EVERYTHING the other Z-80 boards do, plus Z-80 interrupts.
- All new 1983 design incorporates the latest in I.C. technologies.
- Complete documentation included. (User must furnish software)

The Z-80 PLUS turns your Apple into a CP/M based computer. This means you can access the largest body of software in existence. Two computers in one and the advantages of both, all at an unbelievably low price.

- TOTALLY compatible with ALL CP/M software.
- Executes the full Z-80 and 8080 instruction set.
- Fully compatible with microsoft disks (no pre-boot required).

COMING SOON: The Z-80 Plus for the Apple III

PRICE $139.00

Analog to Digital Converter

- 8 Channels
- 8 Bit Resolution
- On Board Memory
- Fast Conversion (.078 ms per channel)
- Eliminates the Need to Wait for A/D Conversion (just PEEK at data)
- A/D Process Totally Transparent to Apple (looks like memory)

The analog to digital conversion takes place on a continuous, channel sequencing basis. Data is automatically transferred to on board memory at the end of each conversion. No A/D converter could be easier to use.

Our A/D board comes standard with 0, 10V full scale inputs. These inputs can be changed by the user to 0, -10V, or -5V, +5V or other ranges as needed.

Information on temperature sensors is given in manual.
The user connector has +12 and -12 volts on it so you can power your sensors.

Accuracy 0.3% Input Resistance 20K Ohms Typ

A few applications may include monitoring and control of ● flow ● temperature ● humidity ● wind speed ● wind direction ● light intensity ● pressure ● RPM ● storage oscilloscope ● soil moisture and many more.

We also manufacture a 16 channel digital input/output board for control applications.

PRICE $129.00

NEW

IIe Only: 80 Column, 64K RAM Card

- Expand your Apple IIe to 128K memory.
- Provides an 80 column text display.
- TOTALLY compatible with ALL Apple software and languages, there are NO exceptions.
- Automatically expands VisiCalc to 95K storage. In 80 columns!
- COMPLETE documentation included. (We don't make you refer to the Apple manual as others do.)
- Uses the same commands as the Apple 80 column board.
- Incorporates the latest high speed, low power I.C. technologies.
- Plugs into the Apple IIe expansion slot.
- Simply the best expansion card for your Apple IIe at any price, offering you phenomenal performance at a very nominal price.

PRICE $149.00

Our boards are far superior to most of the consumer electronics made today. All I.C.'s are in high quality sockets with mil-spec. components used throughout. P.C. boards are glass-epoxy with gold contacts. Made in America to be the best in the world. All products work in APPLE IIe, II and II+. (Except 80 column card.)

Applied Engineering's products are fully tested with complete documentation and available for immediate delivery. All products are guaranteed with a no hassle two year warranty.

All Orders Shipped Same Day
Texas Residents Add 5% Sales Tax
Add $10.00 If Outside U.S.A.
Dealer Inquiries Welcome

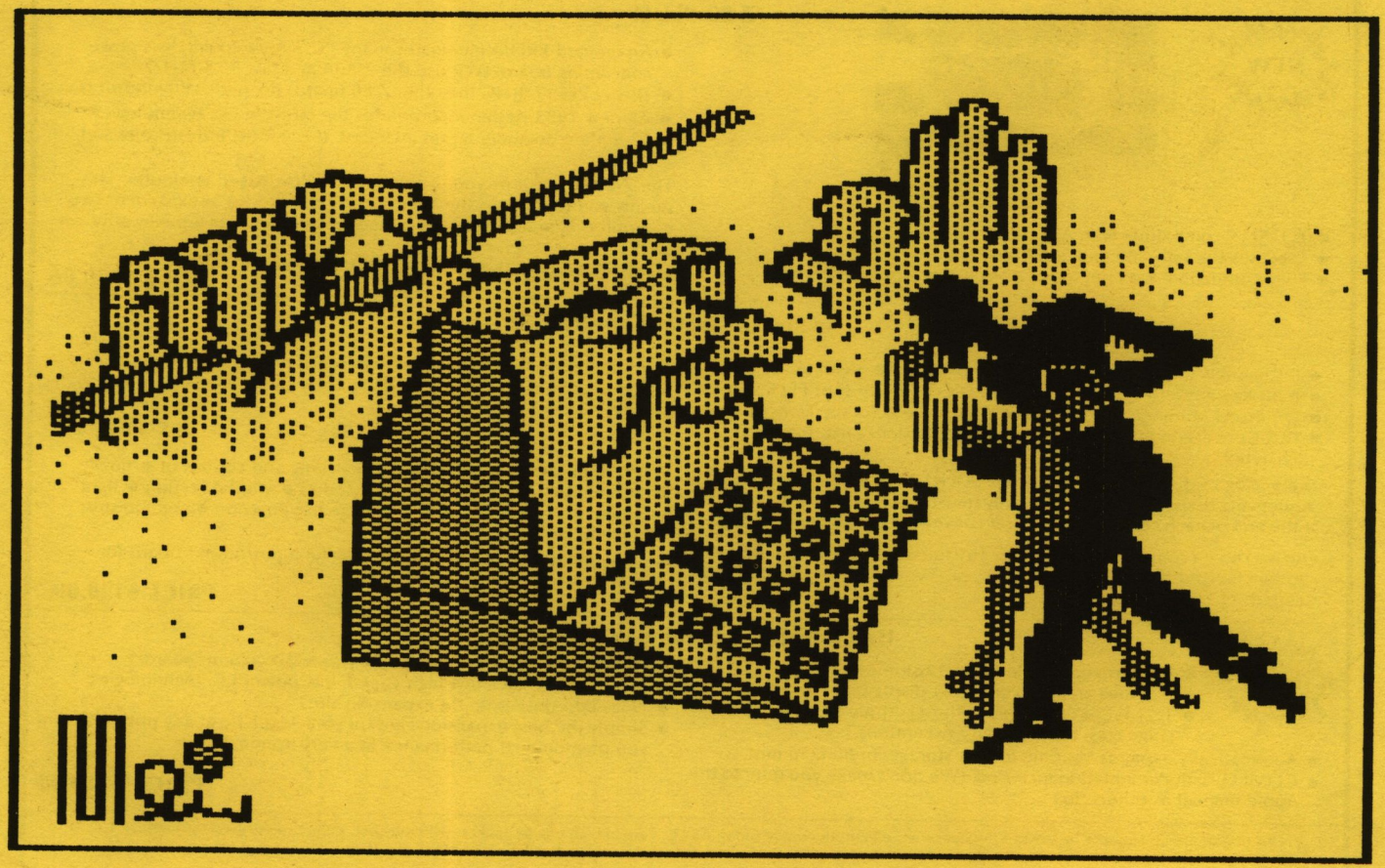
Send Check or Money Order to:
APPLIED ENGINEERING
P.O. Box 470301
Dallas, TX 75247

Call (214) 492-2027
7am to 11pm 7 days a week
MasterCard, Visa & C.O.D. Welcome

SF APPLE CORE
1515 Sloat Blvd., Suite 2
San Francisco, CA. 94132

TIME VALUE

Bulk Rate
U.S. Postage
PAID
Permit #85
El Cerrito, CA 94530

TO:

3121AUG83
DOUGLAS STORKOVICH
921 PINTO CT
WALNUT CREEK CA 94596